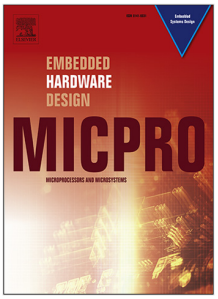
Journal Pre-proof



Exploring the security landscape: NoC-based MPSoC to Cloud-of-Chips

Gaurav Sharma, Georgios Bousdras, Soultana Ellinidou,
Olivier Markowitch, Jean-Michel Dricot, Dragomir Milojevic

PII: S0141-9331(21)00141-1
DOI: https://doi.org/10.1016/j.micpro.2021.103963
Reference: MICPRO 103963

To appear in: *Microprocessors and Microsystems*

Received date : 1 November 2019
Revised date : 21 November 2020
Accepted date : 7 January 2021

Please cite this article as: G. Sharma, G. Bousdras, S. Ellinidou et al., Exploring the security landscape: NoC-based MPSoC to Cloud-of-Chips, *Microprocessors and Microsystems* (2021), doi: https://doi.org/10.1016/j.micpro.2021.103963.

Exploring the Security Landscape: NoC-based MPSoC to Cloud-of-Chips

Gaurav Sharma[a,1], Georgios Bousdras[b], Soultana Ellinidou[a], Olivier Markowitch[a], Jean-Michel Dricot[a], Dragomir Milojevic[b]

Université Libre de Bruxelles, Belgium

[a] *Cybersecurity Research Center*
[b] *BEAMS Department*

Abstract

In this paper, we present a detailed and systematic overview of communication security aspects of Multi-Processor Systems-on-Chip (MPSoC) and the emerging potential threats on the novel Cloud-of-Chips (CoC) paradigm. The CoC concept refers to highly scalable and *composable* systems, assembled not only at system design-time using RTL, like traditional SoC, but also at integrated circuit (IC) packaging time thanks to 3D-IC integration technology. Practical implementation of CoC systems needs to solve the problem of scalable, configurable and secure communication not only between different functional blocks in a single ICs, but also between different ICs in a single package, and between different packages on the same or different PCBs and even between different systems. To boost such extremely flexible communication infrastructure CoC system relies on Software-Defined Network-on-Chip (SDNoC) paradigm that combines design-time configurability of on-chip systems (NoC) and highly configurable communication of macroscopic systems (SDN). This study first explores security threats and existing solutions for traditional MPSoC platforms. Afterwards, we propose SDNoC as an alternative to MPSoC communication security, and we further extend our discussion to CoC systems to identify additional security concerns. Moreover, we present a comparison of SDNoC based approach over existing approaches and discuss its potential advantages.

Email address: gaurav.sharma@ulb.ac.be (Gaurav Sharma)
[1]Cybersecurity Research Center, Université Libre de Bruxelles, Brussels, Belgium

Keywords: Cloud-of-Chips, Security Overview, Software-Defined Networking, Key Management, MPSoC, Network-on-Chip

1. Introduction

Current and next-generation computation systems will combine centralized, and edge computing approach and the end-system will focus on the demands of Internet of Things (IoT). Such applications need versatile multi-layered Multi-Processor Systems-on-Chip (MPSoC) architectures, configurable at design time and run time. Following the trend of SoCs, we propose a novel architecture which can simultaneously satisfy the design and run-time application requirements.

The most important aspect of the Cloud-of-Chips(CoC) architecture is the extensive scalability; the approach can be applied to any system size: from small systems (few dozens of functional blocks) up to large systems with thousands of blocks. Our novel architecture is referred as CoC [1], in analogy to the unlimited scalability of cloud computing paradigm. The CoC concept refers to large amounts of interconnected ICs and IC cores which can have different communication speeds and hierarchy levels. IC cores are using advanced integration technologies such as 2.5D and 3D. The CoC template architecture is a flexible and scalable SoC which can be easily reconfigured, based on the application requirements. The proposed system follows a flexible architecture which can change its characteristics, such as routing logic, transmission path, priorities and IC clustering. The template architecture and computing clusters are coupled at design time while communication scheme and security features can be dealt with at run-time. To describe our proposed architecture, Figure 1 presents a PCB hosting a package of multiple identical ICs where each IC may have many functional IP cores.

In other words, the Cloud-of-Chips is an integration of multiple integrated circuits (ICs) where each IC can be a combination of multiple dies incorporating high and low-performance cores. Moreover, these ICs could be a graphical processing unit (GPU), crypto processor, accelerator or a combination of such IP blocks. The solution is highly scalable and low-cost as it can accommodate any application based on its requirements. The primary motivation is to create subsystems on the integrated system such that different subsystems can work in parallel for better performance. Figure 2 shows a collection of ICs with two subsystems, one high performance subsys-

tem and another high-performance subsystem employing a crypto unit for cryptographic operations.

To enable the concept of CoC and the system configurabilty at the levels mentioned above, we propose to use the idea of Software-Defined Networks (SDN). Each IC integrates a software-programmable controller. All controllers are reporting to the central hardware controller. The two-level of hierarchy enables efficient communication on the IC level as well as the PCB level. The packet forwarding is dealt in an SDN way. The source IP core forwards the packet header to the controller on-board, and the controller sends back the whole sequence of exit ports at each NoC router. The controllers on each IC also maintain flow tables and group tables for outside IC communication. The flow rules include frequently visited paths, and in a case of miss, the packet header is forwarded to the central controller. The central controller has access to global topology view and is responsible for the updates of flow entries on these controllers. Once the flow entry is updated, the header packet is assigned a route and the rest of the packets will follow the same route.

1.1. Potential Applications

The CoC can be a unique platform for a variety of applications. The flexible character of CoC provides the ability to create any SoC as per the primary design specification of the application, by overlapping many steps during design-time, as it exploits existing dies. After the tape-out of the IC, SDNoC, which is an application-specific software-defined network, can provide subsystems into the same system. For example, enabling dedicated paths among selected cores, you can create *high-performance* subsystems, *low-power* subsystems or a blend of both, as per the demands of the application. Besides, in specific applications, such as aviation, medicine or automotive, the embedded accelerators can connect or disconnect on each configured subsystem even at run-time. Since the number of IP blocks are configurable at design time as well as run time, CoC can support several potential applications, discussed briefly as follows:

- Cloud Computing Services/Internet of Things, demands versatile heterogeneous systems comprising of multiple diverse devices. The currently employed systems lack compatibility among heterogeneous devices which inflates the total cost of deployment. Due to high scalability, CoC can be adapted based on application's demands

- Mobile and High Performance Computing, are in a continually evolving stage, challenging hardware innovation which indeed demands embedding plenty of functionalities on the chip. The time-to-market pressure and the implementation cost can be handled with the CoC which reduces the iterator and the time-consuming design flow of the current 2D systems, by using existing functional dies

- Medical Instruments are characterized by heterogeneous architectures which include many processors and accelerators. By combining different CMOS technologies on the same package, CoC makes it feasible to trade-off performance and cost

- Aeronautical/Space and Automotive systems, traditionally use many control modules on dedicated devices to manage the peripheral sensors. All those devices are connected to a central unit which leverages the entire operation. CoC increases the reliability from the security perspective and the performance of the system, by incorporating all the peripheral units on the same package, with 2.5D and 3D stacking integration.

1.2. Motivation and Our Contribution

The target market of MPSoC is growing exponentially with increasing complexity in application design. This opens the attackers more avenues to expand the attack surface of MPSoC. The existing attacks target a specific attack surface (either communication infrastructure, physical or side-channel security) and overlook others and hence, we suggest the overall security should be analyzed collectively. Moreover, recent advancements lead to applicability of SDN on a micro-architectural level, namely SDNoC. However, the idea is explored from the networking viewpoint, but security has been undermined. We present the challenges and benefits of using SDNoC and compare with existing traditional packet routing approaches.
The principal objective in this paper is to explore the security threats on MPSoC platform and extend these issues further to CoC. Firstly we describe the challenges in scaling MPSoC to CoC, and then we explain how this extended platform adds more vulnerabilities. Our discussion can be summarized in the following sequence:

- Identification of threats on MPSoC ranging from secure admission to side-channel attacks

- Protection strategies to execute a sensitive application via security zones
- Propose the SDNoC based security approach and compare with existing approaches
- Analyze the additional vulnerabilities while scaling MPSoC to CoC

Trusted execution environment has widely been investigated and implemented in modern systems. The challenges in safety-critical systems [2, 3] such as avionics, aeronautical and medical equipments might differ than the security critical systems [4, 5]. In this contribution, we explicitly limit our discussion to secure communication on NoC-based MPSoC while executing a secure application.

1.3. Outline of the Paper

Rest of the paper is organized as follows: In Section 2, we revisit related background. In Section 3, we present the security threats on MPSoC platform, and related solutions are discussed in Section 4. The SDNoC security prospects are discussed in Section 5. Some existing security architectures are discussed in Section 6. The security challenges to scale an MPSoC to CoC are described in Section 7, followed by the summary in Section 8.

2. Background

The CoC is an integration of multiple integrated circuits where each IC might be an MPSoC in itself. Moreover, the concept of SDN makes the platform more interesting. The below three subsections will provide an overview of MPSoC used in CoC, the concept of SDN and listing its standard security threats as well.

2.1. Multiprocessor System-on-Chip

System-on-Chips (SoC) are able to include a high volume of transistors on a single chip and, consequently, we can embed a variety of functional blocks into the same system [6]. As the applications start to become more demanding, the evolving SoC increased the number of processors and added more functionalities on the same die. Including more processors or increasing the number of cores, the MPSoCs [7] were capable of leveraging the performance by keeping the power consumption in control. That is possible due to

the ability to execute tasks in parallel while the clock period is low and the power dissipation is down as well.

Besides, the MPSoCs were characterized as a homogeneous architecture which means all the embedded cores have the same architecture such as instruction set, cache hierarchy etc. As the systems start to execute plenty and variety of applications, the need for versatile architectures emerged, targeting the trade-off performance and power consumption. The next generation was to deploy heterogeneous systems where different cores and accelerators are established into the same system. The cores can differ to the architecture, the size they occupy as well as the clock period. Therefore, using a proper software scheduler, we are able to execute different cluster of tasks on different cores, satisfying the application's demands.

Regarding the interconnection, the MPSoC use the widely known bus interconnection (ARM AMBA, IMB CoreConnect) [8]. The buses follow a hierarchical interconnection which can connect single or multi-master controllers to multi-slaves and uses a dedicated controller for the low bandwidth peripheral components. In addition, bridges are used between different performance buses in order to preserve the hierarchy. However, as the complexity of the SoC is increasing rapidly, the traditional buses are unable to meet the performance requirements, particularly on systems with a high number of processing units. As a solution, NoC [9] emerged as state of the art and novel communication infrastructure. At the NoC, the system's processing elements (PE) are interconnected through the interface to a dedicated router, and all the routers are connected usually on a mesh topology or application-specific topology. This interconnection enables simultaneously, the execution on the PE and ensure the Quality-of-Services that requires a complex architecture. A drawback of the NoC is that it uses a static routing algorithm and presents a lack of scalability for a diversity of applications. Studies focus on dynamic NoC, and particular interest surfaced on the SDNoC [10] which is able to program the behaviour of the entire network.

Traditionally, the Heterogeneous MPSoCs are implemented on a single die. Billions of transistors are embedded in a given die area, using diversity of functional blocks. For complex designs, this approach is quite expensive and time-consuming. With advanced integrated circuit (IC) packaging technologies [11], we are able to deploy systems using 2.5D and 3D integration and therefore, deliver complex design architectures with performance and manufacturing cost trade-off. Consequently, we can implement Heterogeneous Systems [12] not only on the architectural design but also, on CMOS ap-

proach by using different technologies on the same package and by exploiting the scalability of SDNoC, the systems are going to be more flexible.

2.2. Overview of Software-Defined Networking

SDN came into the surface to support the future network functions and IoT applications while lowering operating costs by simplifying the hardware, software and management [13]. Although SDN appeared as a research concept in 2008 [14], it quickly gained significant attention from the industry over the past few years. In fact, Google, Facebook, Yahoo, Microsoft, Verizon, and Deutsche Telekom fund Open Networking Foundation (ONF; www.open networkingfoundation.org) adopted the SDN through open standards development.

SDN architecture consists of three main planes, as shown in Figure 3: application, control, and data. The data plane consists of forwarding network equipment, i.e., switches.[2] The control plane contains the controllers which facilitate setting up and tearing down data paths in the network (data plane) according to the requirements of the running applications (application plane). The control plane linked with the data plane via an application programming interface (API), referred to as the south-bound API. If multiple controllers exist, connections among them are called as east and west-bound APIs. The controller-application interface is referred as north-bound API. The goal of SDN is to provide an ability to the users through controllers to control and manage the forwarding plane (hardware) in a network. In other words, SDN exploits the ability to split the data plane (forwarding of the packets) from the control plane (route planning and optimization) [15]. This paradigm provides a view of the entire network and enables global changes without a device-centric configuration on each router separately [16]. Furthermore, the control plane could consist of one or more controllers, depending on the size of the network. In the case of multiple controllers, some reference architectures have already been introduced [17, 18]. The controllers can form a peer-to-peer, high-speed, reliable and distributed network control. The switches from infrastructure plane forward packets among themselves by checking the flow tables that are managed by the controller(s) in the control plane.

[2]For the sake of clarity, it is essential to note that, unlike in traditional networking, the words "switch" and "routers" are referring to the same concept in the field of NOC,.i.e, a packet forwarding entity that interconnects processing nodes and transmit the packets along a pre-defined data path.

As far as the communication between switches and controller, specifically in the south-bound API, there are several communication protocols recently appeared in the literature. Still, one of the most widely used is OpenFlow [14]. In the OpenFlow specification, it is mentioned that the data plane is controlled by providing rules (flows) to the network devices (switches). Each flow entry is an instruction for matching the incoming packets. OpenFlow is basically designed for regular computer networks and therefore a lightweight communication protocol should be designed targeting MPSoC. Moreover, OpenFlow protocol does not enforce security as compulsory, which makes the network open to several attack scenarios [19].

2.3. SDN Security Issues

A number of security analyses have recently been performed [20, 19, 21, 22], which have found that the altered entities or the links between entities in the SDN framework introduce new vulnerabilities, which were not present before SDN. Following the data flow and interaction among SDN entities, Microsoft presents the STRIDE threat model [23] to meet security requirements CIANAA (confidentiality, integrity, authentication, non-repudiation, availability, authorization). STRIDE stands for Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privileges. The attacks are listed as follows:

- Spoofing(Authentication): an attacker masquerades as a legitimate user, by sending packets in order to gain access to the network
- Tampering(Integrity): an attacker attempts to deliberately modify given data from unauthorized transmissions. This could happen when the controller installs flow rules, aiming to modify or falsify data packets or flow counters[23]
- Repudiation(Non-Repudiation): an attacker can send packets to another destination by modifying packet source address. Specifically, repudiation attack occurs when a system does not adopt controls to track user's actions properly, thus permitting malicious manipulation
- Information disclosure(Confidentiality): an attacker has in his possession a piece of information that is not permitted to have. In the context of SDN, this could lead to side-channel attacks intended to reveal extended information about the system

- Denial of service(Availability): an attacker attempts to prevent legitimate users from accessing the service. The DoS attacks are introduced in order to make the system unavailable to receive and transmit data. In the SDN concept, the controller should be aware of the network state on a regular basis in order to apply rules, that makes an SDN base system vulnerable for denial of service attacks [24, 25, 26]

- Elevation of privilege(Authorization): an attacker alters his privilege to have access on the system by performing system operations. In order to perform this attack, an attacker should have access to the controller, which is considered as less critical to happen, due to the proposed use of TLS [27].

The OpenFlow standard describes the use of the Transport Layer Security (TLS) protocol. However, its use is not well enforced [28]. It is written in the specification that "the switch initiates a standard TLS or TCP connection to the controller" which means that the use of TLS is completely optional. In fact, security mechanisms, such as TLS, protect against many attacks, however, the threats should not be overlooked when moving to SDN and OpenFlow [29].

3. Execution of a Secure Application on MPSoC

According to Sepulveda et al. [30], classification of attacks on MPSoC platform includes (a) Denial of Service, (b) change of system behaviour and (c) extract sensitive data from a memory location. The first attack focuses on legitimate resource utilization and on-chip communication security, whereas illegal memory access leads to other types of attacks, such as modifying or accessing sensitive data from shared memory. Another research viewpoint also includes secure admission of application into account [31]. According to them, the execution of a sensitive application comprises at least three following assumptions:

- secure admission of the application to guarantee the object code integrity

- secure access to peripherals and shared memories

- application execution in a protected environment

The overall research on MPSoC focuses mainly on three categories of security threats - secure loading of application software on the MPSoC platform, isolating the resources for secure execution of applications (computation as well as communication shared resources) and secure memory access. The computational resources include processing elements while communication resources cover the shared medium such as AXI, AHB or NoC. The existing industry-based solutions (presented in section 6) focuses on isolating these resources in various means. In most of the existing systems, isolation is achieved by placing a separate dedicated chip for secure functions which is obviously not a cost-efficient solution.

3.1. Secure Admission of Application

The malicious pieces of codes can be inserted into the application code to deploy code injection attack. This compromises the confidentiality and integrity of MPSoC. In order to ensure the secure admission of an application, the integrity of application code must be maintained. The object code integrity is usually achieved with message authentication codes (MAC). While spreading the application on MPSoC, each IP will verify the MAC individually [32]. SipHash-2-4 is one of the recommendations to avoid code injection attack [32]. The pre-stored MAC is compared with computed MAC-over the application memory block. A positive outcome of this match ensures the stored application code is not modified. This verification is possible when MPSoC has a pre-shared key with the application.

Deploying the application code on MPSoCs from external entities without verification is insecure. Therefore, there must be some secure key exchange in place between application provider entity and the MPSoC, Diffie-Hellman key exchange is a well-known public-key cryptography protocol for this purpose. Also, existing standard security protocols TLS/SSL can be used to ensure secure transfer of application data from the source. A demonstration of secure admission of application is presented in Figure 4.

As far as the secure boot is concerned, executable code must ensure confidentiality, integrity and authenticity. For instance, Zync-7000 all programmable SoC which is currently integrated into a wide range of embedded applications including professional cameras, medical endoscope and multifunctional printers, employs crypto primitives such as AES, HMAC and RSA to ensure secure booting [33]. For each partition, the first step is to generate HMAC (in this case SHA-256, a one-way function) followed by AES encryption and then signed using RSA. The same steps are reversed by Zynq devices

to verify the partition.

3.2. Protected Memory Access

In this subsection, we are addressing the security issues related to shared memory access among processing elements while sharing the same address space. Generally, there are two categories of attacks on shared memory access as follows:

- **Extraction of secret information**: reading sensitive data stored in some secure areas such as memory core or other IP cores
- **Hijacking**: These attacks will try to get permission to write some data in a secure memory area so that it can modify the behaviour of the system. These attacks are possible by using buffer overflow or re-configuring the internal registers.

The naive solution in order to protect memory is to analyze each access request. In 2007, Fiorin et al. [34] presented the memory protection aspects in Networks-on-chip infrastructure. This context might have different opinions on implementing a hardware data protection unit (DPU) or a software firewall to filter the packets. Moreover, the requests can be filtered at the initiator IP (requesting IP core) or target IP (memory IP). Both approaches have their pros and cons. The area overhead of individual DPU at each NI is compared with the approach when all the requests are dealt at the memory IP [35]. Considering the DPU at each processor NI and to store the access rights, cache memory is used while in the case of a cache miss, a RAM is there in local memory to check the access rights. The implementation at source IP core causes area overhead while at memory IP, dealing a huge number of access requests is challenging. Furthermore, in order to validate the source, authentication at destination IP is an additional burden.

The core of both approaches is to filter the access request based on its legitimacy. Figure 5(a) and (b) below presents the DPU at memory IP and DPU at individual IP respectively. A DPU filters the packets according to some predefined rules such as:

- Check Source
- Check Destination
- Source Privilege (user, supervisor)

- Allocated memory address block
- Access Type (read, write, none, both)

3.3. Protected Environment for Application Execution

This is the widest attack surface to MPSoC presenting a variety of on-line and off-line possibilities. The system integration phase or supply chain may insert a compromised hardware circuit to gather sensitive information while activated. Also, timing side-channel attacks are also demonstrated on MPSoC during the application execution [36]. The physical attack, such as retrieving the keys from the storage is another applicable attack on such systems. Moreover, attacks due to malware and malicious applications may exploit the resources in an illicit manner.

3.3.1. Hardware Trojan

The challenge to protect NoC IP against hardware Trojans is an immediate necessity. A compromised third-party NoC can invite numerous attacks, including information leakage and denial of service. The NoCs supplied to SoC integrator may have a hardware Trojan embedded in it. The Trojans can evade detection during IP verification as they are designed to remain latent. In order to activate them, a malicious circuit is inserted during the design of the IP block or a malware/malicious program can activate the Trojan at run-time. Usually, they are rare event triggers, activated either by internal or external triggers. There are several contributions presenting hardware Trojans [37, 38], placing them in NoC IP, NI and in DPUs, however, the protection mechanism is mostly implemented in NI. The possible attacks are:

- Snooping of sensitive data (Confidentiality)
- Corrupt the data (Integrity)
- Spoofing (Authentication)
- Denial of service

The protection against hardware Trojans can be a multi-layered approach. The primary focus should be to avoid the activation of the Trojans. Fort-Noc [37] suggests scrambling the data through XOR-cipher based encryptors and decryptors. As the data is ciphered and hence distorted before injected

into the network, the Trojan activation can be avoided. The second layer of security could be the packet authentication and integrity if the first layer protection already failed, i.e. the Trojan is activated. The third layer prevention can be periodic migration of tasks to different IPs such that the source and destination are changing which could make it harder for the attacker to extract sensitive information.

3.3.2. Side-Channel Attacks

Side-channel attacks aim to extract sensitive information by exploiting leakage from cryptographic implementations. The well known side-channel attacks consider pattern analysis for timings, power consumption and electromagnetic emanation. Two orthogonal criteria are discussed depending upon the adversary's physical intrusion into the system - active vs passive and invasive vs semi-invasive vs non-invasive [39]. Timing side-channel attack analyzes cryptographic algorithm execution timings and attempts to retrieve sensitive information [40]. Suppose a processing element during the execution of a secure application, performs some crypto operation such as AES ciphering, the timing analysis can be exploited to recover the secret key used in encryption. Wang and Suh [36] identified the need for timing channel protection for on-chip networks.

A close variation to timing attack is a cache-timing attack where side-channel information is gathered by measuring cache access timings. In 2016, a practical cache-timing attack was launched on NoC using Prime+Probe technique [41]. By running spy process and victim process on different IP cores sharing a cache memory, NoC traffic throughput is analyzed. The attack targets the communication between an ARM Cortex-A9 core and a shared cache memory.

- The attacker injects the packets and closely monitors the attacker throughput. The collision between sensitive traffic and attacker traffic is the key success point
- The throughput of the infected IP reveals the access pattern and the volume of communication over the sensitive path
- In 76 iterations, 12 bytes of AES 16 byte key was revealed, rest Brute force can help

Other hardware-assisted side-channel attacks such as circuit's timing, power consumption and electromagnetic emanations have been carried out and fur-

ther analyzed using advanced statistical methods. The most common countermeasures for side-channel attacks are random masking techniques, insertion of dummy code and power consumption randomization. Additionally, fault-injection attacks are also feasible. White light, laser beams, voltage glitches, and temperature control are possible manipulation means to perform fault attacks.

3.3.3. Computation and Communication Issues

The threat model for computation and communication protection mainly covers denial of resources. These attacks aim to waste the system resources in several ways. The possible attacks under this category are replay, incorrect path, deadlock and livelock.

- Replay: The attacker may replay the packets to exploit resources such as network-on-chip bandwidth
- Incorrect path: The attacker may forward the packets to incorrect paths so that they will never reach their destination
- Deadlock: The attacker position himself in such a way that the NoC routers are waiting for other neighbouring routers to get available
- Livelock: The attacker forwards the packets in such a loop that they will live forever in the network but never reach the destination

The naive solution to protect the system is to reserve the computation and communication resources exclusively for all security-critical applications. However, this is extremely inefficient in terms of performance. The solution should allow reasonably shared resources without leaking any sensitive information. The existing solutions mainly focus on creating security zones, described later in following section. The below section categorizes these security zone solutions based on the approach used.

4. Protection through Security Zones

In order to execute a sensitive application on an MPSoC platform, the operating system allocates some of the trusted IP cores. The distribution of threads to different IP cores forces them to exchange sensitive data. The plaintext communication on the NoC leads to the breach of information. The wrapping of these IP cores into a physical zone is known as a security

zone. In other words, a security zone is a physical or logical group of IPs protecting the sensitive application from any leakage. The security zones are categorized into continuous and disrupted security zones depending upon the availability of IPs. A continuous security zone when all the participating IPs are physically close to each other while the disrupted zone may have a collection of multiple continuous zones. The protection in disrupted security zone is more challenging than continuous security zones. There can be several ways to achieve secure communication among IPs on MPSoC. Some of them (not independent of each other) can be listed as follows:

- Creation of security zones and protecting them via firewalls or wrappers around it
- Secure packet routing among these zones (encapsulating the route within a security zone)
- Secure communication among these zones with an appropriate key agreement approach

Moreover, there are some situations when a security zone needs to be modified during run-time:

- a new application is mapped on the system
- a task is migrated between IPs on MPSoC
- under special operating conditions such as attack

The challenge is to reshape a security zone during application execution. These security zones are dynamic secure zones that may be continuous or disrupted. Recall that a dynamic security zone enables adding and removing IP cores at run-time.

4.1. Firewall Protection

Firewalls are the most common solution for creation of security zones. However, this solution has its own limitations. A firewall can protect only a continuous security zone and therefore is not effective for disrupted security zones. The idea of a dynamic firewall was first addressed by [42].

The security policy implemented in these firewalls is integrated in network interface (NI), and it filters the data forwarded and received by the IP cores

of a security zone. These firewalls can either be static or dynamic. The static firewall has fixed access rules [35] while the re-configurable firewalls update the security policy regularly [43, 44, 45]. It is assumed that the IP cores inside a security zone trust each other. Figure 6 shows firewall in NI and similar to the DPU, the firewalls filter source ID, destination ID, allowed address range, permissions and finally, whether the transaction is valid or not.

A similar approach to firewalls is implementing Wrappers. A wrapper discards all the incoming and outgoing traffic to a port. This approach finds a suitable number of IP cores at a continuous location, enables the wrappers on the boundary IPs and creates a secure zone. If IP cores are not available at a continuous location, it migrates the tasks to some other idle IPs. All the traffic crossing the security zone need to be rerouted. Therefore, secure routing is an additional measure required for this purpose.

Furthermore, in the presence of a hardware Trojan, the traffic should be encrypted and authenticated to avoid any compromise. When an attacker performs a timing attack, the firewall can identify the increase in bandwidth in the attacker NI. Similarly, a sequence of similar requests can hint for the identification of attacker. If the embedded firewall is only for the purpose of attack detection, the additional preventive measure needs to be adopted, such as security-aware routing, described below.

4.2. Security-Aware Routing

The execution of any sensitive application attempts to spread the application on multiple IPs for better efficiency. These IP cores need to communicate frequently. As the IP cores involved in security zone are trusted to each other, a continuous security zone can be protected with firewalls on the boundary. Unfortunately, if the IPs are not adjacent to each other, they need to exchange the messages on the communication fabric. The infected IP on this routing path can extract some sensitive information. This sensitive path communication by disruptive security zones must be confidential and no infected IP must be able to extract any information from this. The confidentiality or integrity of the communicated data is provided with the encryption and hashing of the data. Whenever a malicious IP is on a sensitive path, an adaptive routing can be a viable solution to find another path. Figure 7 assumes a malicious IP and therefore, modified routing is adapted at run-time. Let S and D be the two IPs running threads of the same task and they need to exchange some data, which is generally plaintext (unencrypted). Here we

assume S and D as the source and destination IPs respectively. Also, we assume the traditional XY/YX routing in place. The dotted arrow should be the usual path if there is no infected IP on this route. Due to the presence of infected IP on this route, the adaptive routing will suggest another route with minimal changes, and also it should be free from deadlock, livelock and starvation.

The primary objective is to maximize the encapsulation of sensitive path inside a security zone. In 2016, Fernandes et al. [42] presented a security-aware routing approach for NoC based MPSoCs. Their approach uses a segment based routing (SBR) algorithm to find the route between source IP and destination IP. However, the solution has a performance impact, but it avoids those routes which include routers attached with infected IPs. The existing security zone solutions attempt to avoid the routers attached with malicious IP cores. The adaptive routing at this point may choose another route which is longer and more risk-sensitive. Sepulveda et al. [46] introduced the concept of risk-aware routing. The risk factor depends on the malicious activities of the infected IP, and whenever the security rules are violated, a notification is communicated to the security manager. The presented solution - global risk-aware NoC architecture (GRANOC) searches a new route as the risk exceeds the threshold value.

Later, Sepulveda et al. [47] proposed another security-aware routing approach in zone-based MPSoC. The region-based routing (RBR) is used at design time while at run time non-minimal odd-even (NOE) routing is followed. The routing decision must be driven by the risk value of the hop.

The existing approaches commonly avoid DoS attack. The basic assumptions in such proposals are as follows:

- The NoC is secure
- The participating IP cores trust each other
- Attacker knows the MPSoC mapping strategy
- Attacker is aware about the routing algorithm
- Attacker IP is located inside a sensitive path

4.3. Intrazone Key Agreement on MPSoC

In the previous discussion, the primary focus was to create an envelope surrounding the IP cores, involved in security zone. The firewall is installed to

filter the traffic based on security rules. The security policy can be upgraded during run time. The firewall-based approach is limited to the creation of continuous security zones only. These IPs become unreachable for other IP cores, and hence it degrades the overall system performance. On the contrary, the disruptive security zones are forced to communicate on a sensitive path. The communication among these zones must be protected to maintain confidentiality and integrity. Moreover, a session key is needed to enable such secure communication among these IP cores. Figure 8 illustrates a crypto module in NI which can generate random numbers, convert plaintext into ciphertext, maps a random length string to a fixed-length string via hashing and a key manager can be there to store the derived session key. Moreover, it can also include a protocol accelerator which can substantially enhance task efficiency. Such crypto cores can be essential when sensitive applications are exchanging instructions on insecure network-on-chip infrastructure.

To derive the session key among these IP cores, the initial solutions attempt to achieve pair-wise communication security [48, 49]. However, they do not support group-wise confidential communication. In 2014, Sepulveda et al. [45] presented elastic security zones for 3D-MPSoC with group-wise shared secret key establishment protocols. Later, another feasible solution was suggested in [50], which employs the use of hybrid group key agreement protocol. As a common standard in Internet, the asymmetric cryptosystem is used for session key establishment and a symmetric algorithm for data encryption thereafter. Their work [50] implemented two different approaches. The first approach is based on key pre-distribution and assumes that a pool of keys is already distributed to IPs at design time and a key is negotiated at run time. The other approach counts on Diffie-Hellman and derives the shared secret as a function of secrets of all the participating IP cores. The major limitation of these approaches is a lack of scalability and efficiency.

In a further enhancement, Sepulveda et al. [51] implemented three hierarchical group-wise key agreement protocols. A comparative analysis between flat and hierarchical key agreement protocols ensure the superiority of the latter one. In their approach, all the IP cores are assumed to store a private key at design time and a global manager (GM) is supposed to keep all the private keys to securely communicate with them. The idea to store all the private keys at a single point (at GM) makes the whole architecture compromised if GM is compromised. In 2017, Sharma et al. [52] presented a lightweight group key agreement protocol which can be utilized in order to form such security zones. From the above discussion, it is easy to perceive

that intrazone security has been addressed well but interzone security still needs exploration. To the best of our knowledge, the only contribution [53] which addresses the intra and interzone communication security simultaneously.

In order to recap the Sections 3 and 4, we present the following Table 1 which summarizes the MPSoC existing threats and their countermeasures. Analyzing these threats, we conclude that multiple protection mechanisms need to be applied to secure the MPSoC platform. In the following section, we explore the SDNoC potential for MPSoC security.

5. SDN as a Potential Solution

In the following subsections, we briefly discuss the state of the art of SDNoC based solutions.

5.1. SDNoC: Networking Viewpoint

The SDN based networking strategy enables the communication between any two ICs on CoC. To the best of our knowledge, there is no existing literature for CoC. However, there is limited literature available in NoC-based MPSoC which includes SDN as a packet routing approach. SDNoC is a NoC communication paradigm rather than a specific design and implementation, presented for first time in 2014 by Cong et al. [54]. The basic architecture of SDNoC design can be referred from Figure 9. According to the authors of [54, 55], SDNoC could be adaptable for future SoCs owing to its advantages: 1) it reduces the hardware complexity, 2) it has high re-usability and 3) it has flexible management of communication policies. Precisely with the enhanced flexibility achieved through the SDN-based approach, it is possible to accommodate application-specific routing technique within the software-based controller, well addressed by Ellinidou et al. [55]. Moreover, the power consumption of the SDNoC-scheme is proportional to the traffic due to its event-driven nature[56]. In case of no traffic crossing a router, the router could enter the idle (i.e., low-power) state and when there are no new requests, the controller does not need to perform any computation.

With its programming flexibility and obvious advantages in performance and energy consumption, SDNoC recently attracted the attention of researchers. The authors of [54] introduced the SDNoC architecture where the control plane is deployed as a distributed entity at each router, however, this is contrary to SDN philosophy because both planes are placed inside the

router. The presented approach was compared with static XY and dynamic DyAD routing with traditional NoC architecture.

Afterwards, the authors in [57] applied SDN principles in order to propose a SDNoC architecture. This architecture is focused on abstraction layers and interfaces that permit its deployment in a modular fashion and it has the potential to overcome the NoC management problems in the many-core era. However the authors propose an architecture without providing enough details about the security aspects or the communication protocols. Another interesting contribution of the same authors is presented in [58], where they evaluate the SDNoC architecture among the processing elements(PEs) in a many-core system with SystemC simulator, focusing on the configuration time, delay and throughput of their architecture.

Scionti et al. [59] used the SDN architecture in order to explore dynamic changes in the network topology, each PE has specific instructions to control the network topology by software, including switching off the links which are not used. In 2017, Berestizshevsky et al. [60], presented a novel NoC architecture, called SDNoC, based on a hybrid hardware/ software approach. Their approach implements a software based centralized Network Manager (NM), executed on a dedicated core (refer Figure 10). The NM allocates the route and switches forward the packets without storing them. Also, the switches do not maintain any routing tables. An improved solution is recently introduced by Fathi and Kia in [61] where all the routers need not to reach the controller. The router attached to the source IP core sends the packet header to controller and controller provides a sequence of exit ports at each router on the route. All the other intermediate routers check the packet header and forwards the packet to already mentioned exit port.

Ruaro et al. [62] propose a SDN circuit switching (CS) infrastructure for many-cores. This approach enabled to design a simple MPN for CS, through configurable CS routers based on Elastic-Buffers. The main goal of this contribution was to establish CS for real-time application flows with run-time support. Furthermore, with the help of A clock-cycle accurate RTL model, SDN is compared to the parallel-probing (PP) method. The same authors in [63], presented the pros and cons of the SDN paradigm, by simulating in a cycle-accurate many-core model, filling a hole in the literature by proposing a generic SDN architecture and addressing hardware and software implementation details.

Recently, Ellinidou et al. [55] introduced a novel SDNoC communication protocol standard called MicroLET, specifically designed for future

chiplet-based systems. Precisely the authors presented the SDNoC integration within chiplet-based systems by focusing on networking aspects of NoC for intra-chiplet communication, inspired by large scale networks, and with a new routing approach. An implementation of the SDNoC and an evaluation of the proposed routing algorithm compared to the XY and the Odd-Even algorithms within different traffic scenarios is also presented. Through the evaluation of the MicroLET protocol, the authors claimed that it could be a good candidate for the future chiplet-based systems.

5.2. Proposed SDNoC Security Prospect

To the best of our knowledge, there is no existing analysis of SDN based approach from security point of view. Recall that, in the existing techniques for MPSoC security such as firewall, security-aware routing or key management, packet delivery is determined with packet switching routing approach whereas SDNoC solution prefers circuit switched routing. There are two types of security zones mentioned in literature, continuous and disrupted security zones. Since SDNoC can restrict routing using circuit switching, continuous security zones can be easily formed without applying expensive key agreement approach. Moreover, disrupted security zone is also straightforward unless a hardware Trojan is inserted during supply chain. In the presence of malicious IP, other approaches like firewalls or wrappers can not create disrupted security zones alone. Therefore, an additional feature such as secure routing or key agreement is needed. As discussed earlier, secure routing can avoid this malicious IP, and key agreement can encrypt the communication and makes the data unusable for malicious IP. Table 2 presents the secure zone possibility with different techniques.

The SDNoC approach can directly benefit in following aspects:

- resources can be reserved in such a way that it can avoid communication collisions between sensitive and non-sensitive traffics. Therefore, SDNoC provides a dedicated path which implies no timing attack
- no congestion issues, no communication resource sharing implies no livelock, deadlock or incorrect path
- the security-aware routing approach is also dependent on finding an alternate route which can avoid malicious IPs on the route to destination. With SDNoC approach, security-aware routing can be easily achieved.

A comprehensive view of MPSoC security threats, relevant attacks and their countermeasures is presented in Figure 11. Similar to other existing security approaches focusing on separating computation and communication resources, the SDNoC approach also attempts to create secure zone countering the DoS attack. Moreover, avoiding collision between sensitive and non-sensitive data SDNoC approach can additionally avoid timing attack as well. The transition from packet to circuit switching makes this solution more interesting. Here it is clear from Figure 11 that all these routing based security solutions are worth whenever there is no hardware Trojan. The key management is the only solution in case a hardware Trojan is present. The encrypted communication enables the protection against such embedded Trojans. Table 3 compares the existing approaches with SDNoC and ensures that a combination of SDNoC and key management is an appropriate solution.

Counting only the benefits of SDNoC approach does not qualify it as a secure solution. The best point to counter all the above advantages is, the compromise of SDN controller will lead to the whole system compromise. Moreover, the controller is a bottleneck for serving route requests from all the IPs and it would be worse in case of a large number of IPs. On one side, it removes routing tables from each router, but additional communication links are needed among routers and controller, which may lead to increased area overhead.

6. Existing Security Architecture

The industry is promptly adopting more advanced security architectures for improved security on MPSoCs. ARM TrustZone [64] and Intel's SGX [65] are two well-established examples. ARM's TrustZone provides an abstraction of secure and normal world where the secure world runs security critical applications providing confidentiality and integrity to the system while the normal world deals with regular functionality. The communication is restricted between the secure and normal world. The TrustZone technology is being integrated into Cortex processors fulfilling the security need of billions of applications.

Intel's SGX (security guard extensions) protects the security critical applications using enclaves. An enclave is a special cache memory region to hold critical instructions and private data. The applications running inside enclaves are isolated from other applications such as operating system, virtual machine or hypervisor. The trusted part of the application runs on

the enclave, and returns the output to the untrusted part in encrypted form. This hardware-supported trusted execution environment can perform remote attestation to ensure, the integrity of enclave and it is being executed on a legitimate platform. The third-party compares the cryptographic hash to verify the consistency of enclave.

Xilinx Zynq Ultrascale+ MPSoC is extending the above protection to their 3D ICs platform which uses stacked silicon interconnect technology. It includes multiple processing units, four Cortex^{TM}-A53 cores, two Cortex-R5 cores, a platform management unit (PMU), configuration security unit (CSU) and a user-specified number of Microblaze^{TM} processors in the programmable logic [66]. The parallel execution of multiple applications can create interference among subsystems. Isolation is needed in order to run them securely. Similar to ARM, the TrustZone on this platform defines two worlds - secure and non-secure world. The separate allocation of CPUs on Zynq Ultrascale+ provides the isolation between the secure and non-secure world. The accessibility of memory blocks is also restricted for the non-secure world.

The CoC thanks to the configurability, it is capable of adopting all the above security architectures. CoC may host secure cores and create trusted zones like the ARM TrustZone with the help of SDNoC, which enables a configurable interconnection among resources. In addition, CoC exploits the 3D ICs technology like Ultrascale+ MPSoC, and it can generate highly scalable flexible systems with High-Bandwidth-Memory and use layered cache memory in order to increase the density and the bandwidth of transactions, or to provides trusted memory regions like the Intel's SGX. However, shifting from the monolithic MPSoC to CoC - a 3D ICs heterogeneous CMOS system, we encounter new security challenges where we attempt to analyse in the following section.

7. Security Challenges to Scale MPSoC to CoC

A fine-grained collection of reusable blocks (a conventional IC which can be an MPSoC in itself), to fulfil the requirements of a particular application suite can be arranged on a PCB in 2.5D or 3D integration. Besides the integration challenges, our focus here is to analyze security threats while SDN practice is in consideration. Whenever a SoC vendor integrates multiple ICs together, these blocks always have a trust level depending upon the supplier or license owner. They also have their internal routing as well as

other performance metrics. Therefore, if we consider CoC as an assembly of multiple building blocks which are complete in itself, we only need to focus on communication among these blocks. Therefore, it is reasonable to embed an SDN controller connecting these ICs, and this controller is managing the flow entries for routing on each IC.

The security issues on CoC platform can be addressed in a multilayer hierarchical manner. The lowest layer is the IC level where an MPSoC platform is integrated to run multiple applications simultaneously. The previous sections describe the security threats on MPSoC platform. Since, CoC itself comprises multiple MPSoCs, at least the above-discussed security threats are directly applicable here too. Moreover, the presented CoC platform manages the subsystems to run various applications on different ICs utilizing SDN-based switches and controllers. Therefore, some additional security threats need to be appended in the above list.

The second-highest layer is to ensure security among ICs, which primarily depends on how they are structured in a PCB design. One of the primary security benefits in 3D IC stacking is, it can hide the details of circuit design and therefore reverse engineering is difficult. The vertical communication and the interposer plane that are used for the 3D integration emerge unique security threats. As CoC will have heterogeneous 3D stacking where dies are supplied from different manufacturers, a malicious die can perform attacks to extract secret keys. Indeed, the overall security depends upon the weakest die.

The authors [67] study a side-channel attack, exploiting the high-bandwidth that enables the vertical communication using Through-Silicon-Via (TSV) in order to utilize the random eviction which is a costly measurement method on 2D ICs. In addition, the TSVs can be used as a hardware Trojan, by modifying the process parameters (wire height - dielectric thickness) [68]. Therefore, the untrusted manufacturer can affect the performance and reliability of the 3D ICs. Moreover, the silicon interposer needs more concrete security analysis, as it is easy to host a transistor switch here [69] and further enables secondary connectivity between two TSVs. The transistor switch operation is possible through a thermal sensor, and it can share data to a neighbouring die.

Following the SDN strategy, we should also address communication security among switches and controllers. From the communication perspective, the SDN approach divides the overall communication architecture into three planes: data plane, control plane and management plane. A malicious ap-

plication running on one IC can severely impact the performance of CoC. If the switch-controller communication is intercepted, the attacker might store all history command and utilize them later for a specific purpose. Additionally, the switch-controller communication can be flooded with fake requests and controller is unable to handle them and become unresponsive. On the other hand, switches can be flooded with wrong flow table entries. Our CoC is a reactive SDNoC based design where a route request is served by the controller by updating the flow entries on local switch. If the controller's routing application gets infected or acquired from some untrusted source, it can bring down the entire controller.

Since the CoC is highly scalable infrastructure, a hierarchy of controllers may be needed to manage the communication among ICs or even PCBs. The management of multiple controllers might be an issue when multiple ICs are there to handle complex applications. Each controller receives routing updates from other controllers. For two controllers at the same level but in different domains, some security issues can be additional. They may not have the same trust level. There might be issues on cross-domain trust relationship between switches and controllers. Moreover, if the SDN controllers are hierarchical, one SDN controller could be seen as a SDN switch from the upper-level SDN controller.

The highest layer of security could be the PCB-PCB communication which completely depends on the medium among them. For example, in a smart home scenario, multiple PCBs such as reporting sensors, actuators, gateway and a cloud service provider collecting all information and aggregating these reports to concerned authority through some application. This communication can be short-range to long-range and therefore different communication techniques requiring different security suites.

8. Summary

In this paper, we presented a detailed security review for NoC-based MPSoC platform and extended our discussion to identify additional threats while integrating these MPSoCs in a 2.5D or 3D architecture and leveraging SDN flexibility for efficient practice. Based on the related literature, we collected all the relevant threats applicable to MPSoC and compared with the proposed SDNoC based solution. In order to address all the NoC security threats, we need to apply multiple techniques simultaneously. However, the SDNoC based approach can provide an efficient alternative with a fine-

grained access control. The hardware Trojan attack, which can make the data inaccessible to legitimate IPs needs special attention. The encrypted communication is the only means to avoid such an attack. Therefore, some key management strategy must be enabled in addition to conventional solutions. In our findings, the SDNoC based solution with a key management approach is reasonably effective avoiding all the attacks. Furthermore, we realized that a more focused investigation is needed in order to extend our security evaluation from the monolithic IC level MPSoC to CoC communication security issues, which we consider as our future work.

References

[1] G. Bousdras, F. Quitin, D. Milojevic, Template architectures for highly scalable, many-core heterogeneous soc: Cloud-of-chips, in: 2018 13th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2018, pp. 1–7. doi:10.1109/ReCoSoC.2018.8449383.

[2] C. El Salloum, M. Elshuber, O. Höftberger, H. Isakovic, A. Wasicek, The across mpsoc–a new generation of multi-core processors designed for safety–critical embedded systems, Microprocessors and Microsystems 37 (8) (2013) 1020–1032.

[3] A. Burns, R. Davis, Mixed criticality systems-a review, Department of Computer Science, University of York, Tech. Rep (2013) 1–69.

[4] J. Windsor, K. Eckstein, P. Mendham, T. Pareaud, Time and space partitioning security components for spacecraft flight software, in: 2011 IEEE/AIAA 30th Digital Avionics Systems Conference, IEEE, 2011, pp. 8A5–1.

[5] J. Alves-Foss, P. W. Oman, C. Taylor, W. S. Harrison, The mils architecture for high-assurance embedded systems, International journal of embedded systems 2 (3-4) (2006) 239–247.

[6] R. Saleh, S. Wilton, S. Mirabbasi, A. Hu, M. Greenstreet, G. Lemieux, P. P. Pande, C. Grecu, A. Ivanov, System-on-chip: Reuse and integration, Vol. 94, 2006, pp. 1050–1069. doi:10.1109/JPROC.2006.873611.

[7] W. Wolf, A. A. Jerraya, G. Martin, Multiprocessor system-on-chip (mpsoc) technology, Vol. 27, 2008, pp. 1701–1713. doi:10.1109/TCAD.2008.923415.

[8] Amba trademark license, http://arm.com/about/trademarks/arm-trademark-list/AMBA-trademark.php (1996).

[9] S. Kumar, A. Jantsch, J. . Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, A. Hemani, A network on chip architecture and design methodology, in: Proceedings IEEE Computer Society Annual Symposium on VLSI. New Paradigms for VLSI Systems Design. ISVLSI 2002, 2002, pp. 117–124. doi:10.1109/ISVLSI.2002.1016885.

[10] R. Sandoval-Arechiga, R. Parra-Michel, J. L. Vazquez-Avila, J. Flores-Troncoso, S. Ibarra-Delgado, Software defined networks-on-chip for multi/many-core systems: A performance evaluation, in: 2016 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), 2016, pp. 129–130. doi:10.1145/2881025.2889474.

[11] S. W. Yoon, B. Petrov, K. Liu, Advanced wafer level technology: Enabling innovations in mobile, iot and wearable electronics, in: 2015 IEEE 17th Electronics Packaging and Technology Conference (EPTC), 2015, pp. 1–5. doi:10.1109/EPTC.2015.7412320.

[12] R. Kumar, D. M. Tullsen, N. P. Jouppi, P. Ranganathan, Heterogeneous chip multiprocessors, Vol. 38, IEEE Computer Society Press, Los Alamitos, CA, USA, 2005, pp. 32–38. doi:10.1109/MC.2005.379.
URL https://doi.org/10.1109/MC.2005.379

[13] K. Benzekki, A. El Fergougui, A. Elbelrhiti Elalaoui, Software-defined networking (sdn): a survey, Security and communication networks 9 (18) (2016) 5803–5833.

[14] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, OpenFlow: enabling innovation in campus networks, ACM SIGCOMM Computer Communication Review 38 (2) (2008) 69–74.

[15] F. Hu, Q. Hao, K. Bao, A survey on software-defined network and openflow: From concept to implementation, IEEE Communications Surveys & Tutorials 16 (4) (2014) 2181–2206.

[16] S. Das, G. Parulkar, N. McKeown, P. Singh, D. Getachew, L. Ong, Packet and circuit network convergence with openflow, in: Optical Fiber Communication Conference, Optical Society of America, 2010, p. OTuG1.

[17] K. Phemius, M. Bouet, J. Leguay, Disco: Distributed multi-domain sdn controllers, in: 2014 IEEE Network Operations and Management Symposium (NOMS), IEEE, 2014, pp. 1–4.

[18] A. Krishnamurthy, S. P. Chandrabose, A. Gember-Jacobson, Pratyaastha: an efficient elastic distributed sdn control plane, in: Proceedings of the third workshop on Hot topics in software defined networking, ACM, 2014, pp. 133–138.

[19] H. Zhang, Z. Cai, Q. Liu, Q. Xiao, Y. Li, C. F. Cheang, A survey on security-aware measurement in sdn, Security and Communication Networks 2018 (2018).

[20] R. Kloti, V. Kotronis, P. Smith, Openflow: A security analysis, in: Network Protocols (ICNP), 2013 21st IEEE International Conference on, IEEE, 2013, pp. 1–6.

[21] S. Scott-Hayward, G. O'Callaghan, S. Sezer, SDN security: A survey, in: Future Networks and Services (SDN4FNS), 2013 IEEE SDN For, IEEE, 2013, pp. 1–7.

[22] E. B. Eskca, O. Abuzaghleh, P. Joshi, S. Bondugula, T. Nakayama, A. Sultana, Software defined networks' security: An analysis of issues and solutions.

[23] S. Hernan, S. Lambert, T. Ostwald, A. Shostack, Threat modeling-uncover security design flaws using the stride approach, MSDN Magazine-Louisville (2006) 68–75.

[24] Q. Yan, F. R. Yu, Distributed denial of service attacks in software-defined networking with cloud computing, IEEE Communications Magazine 53 (4) (2015) 52–59.

[25] B. Wang, Y. Zheng, W. Lou, Y. T. Hou, Ddos attack protection in the era of cloud computing and software-defined networking, Computer Networks 81 (2015) 308–319.

[26] J. Ashraf, S. Latif, Handling intrusion and ddos attacks in software defined networks using machine learning techniques, in: Software Engineering Conference (NSEC), 2014 National, IEEE, 2014, pp. 55–60.

[27] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, N. Rao, Are we ready for sdn? implementation challenges for software-defined networks, IEEE Communications Magazine 51 (7) (2013) 36–43.

[28] O. N. Foundation, OpenFlow Switch Specification Version 1.5.1 (Protocol version 0x06), 2015.

[29] M. Antikainen, T. Aura, M. Särelä, Spook in your network: Attacking an sdn with a compromised openflow switch, in: Nordic Conference on Secure IT Systems, Springer, 2014, pp. 229–244.

[30] M. J. Sepulveda, J.-P. Diguet, M. Strum, G. Gogniat, Noc-based protection for soc time-driven attacks, IEEE Embedded Systems Letters 7 (1) (2015) 7–10.

[31] L. L. Caimi, V. Fochi, E. Wachter, D. Munhoz, F. G. Moraes, Activation of secure zones in many-core systems with dynamic rerouting, in: Circuits and Systems (ISCAS), 2017 IEEE International Symposium on, IEEE, 2017, pp. 1–4.

[32] S. P. Azad, B. Niazmand, G. Jervan, J. Sepulveda, Enabling secure mpsoc dynamic operation through protected communication, in: 2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS), IEEE, 2018, pp. 481–484.

[33] L. Sanders, Secure boot of zynq-7000 all programmable soc, Application note XAPP1175 (v1. 0), Xilinx (2013).

[34] L. Fiorin, C. Silvano, M. Sami, Security aspects in networks-on-chips: Overview and proposals for secure implementations, in: Digital System Design Architectures, Methods and Tools, 2007. DSD 2007. 10th Euromicro Conference on, IEEE, 2007, pp. 539–542.

[35] L. Fiorin, G. Palermo, S. Lukovic, V. Catalano, C. Silvano, Secure memory accesses on networks-on-chip, IEEE Transactions on Computers 57 (9) (2008) 1216–1229.

[36] Y. Wang, G. E. Suh, Efficient timing channel protection for on-chip networks, in: Networks on Chip (NoCS), 2012 Sixth IEEE/ACM International Symposium on, IEEE, 2012, pp. 142–151.

[37] D. M. Ancajas, K. Chakraborty, S. Roy, Fort-nocs: Mitigating the threat of a compromised noc, in: Proceedings of the 51st Annual Design Automation Conference, 2014, pp. 1–6.

[38] H. M. Wassel, Y. Gao, J. K. Oberg, T. Huffmire, R. Kastner, F. T. Chong, T. Sherwood, Surfnoc: a low latency and provably non-interfering approach to secure networks-on-chip, ACM SIGARCH Computer Architecture News 41 (3) (2013) 583–594.

[39] R. Spreitzer, V. Moonsamy, T. Korak, S. Mangard, Systematic classification of side-channel attacks: A case study for mobile devices, IEEE Communications Surveys & Tutorials 20 (1) (2017) 465–488.

[40] P. C. Kocher, Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems, in: Annual International Cryptology Conference, Springer, 1996, pp. 104–113.

[41] C. Reinbrecht, A. Susin, L. Bossuet, G. Sigl, J. Sepúlveda, Side channel attack on noc-based mpsocs are practical: Noc prime+ probe attack, in: 2016 29th Symposium on Integrated Circuits and Systems Design (SBCCI), IEEE, 2016, pp. 1–6.

[42] R. Fernandes, C. Marcon, R. Cataldo, J. Silveira, G. Sigl, J. Sepúlveda, A security aware routing approach for noc-based mpsocs, in: Integrated Circuits and Systems Design (SBCCI), 2016 29th Symposium on, IEEE, 2016, pp. 1–6.

[43] R. Fernandes, B. Oliveira, J. Sepúlveda, C. Marcon, F. G. Moraes, A non-intrusive and reconfigurable access control to secure nocs, in: 2015 IEEE International Conference on Electronics, Circuits, and Systems (ICECS), IEEE, 2015, pp. 316–319.

[44] M. D. Grammatikakis, K. Papadimitriou, P. Petrakis, A. Papagrigoriou, G. Kornaros, I. Christoforakis, M. Coppola, Security effectiveness and a hardware firewall for mpsocs, in: 2014 IEEE Intl Conf on High Performance Computing and Communications (HPCC), IEEE, 2014, pp. 1032–1039.

[45] J. Sepulveda, G. Gogniat, D. Flórez, J.-P. Diguet, C. Zeferino, M. Strum, Elastic security zones for noc-based 3d-mpsocs, in: Electronics, Circuits and Systems (ICECS), 2014 21st IEEE International Conference on, IEEE, 2014, pp. 506–509.

[46] J. Sepulveda, D. Flórez, R. Fernandes, C. Marcon, G. Gogniat, G. Sigl, Towards risk aware nocs for data protection in mpsocs, in: Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2016 11th International Symposium on, IEEE, 2016, pp. 1–8.

[47] J. Sepulveda, R. Fernandes, C. Marcon, D. Florez, G. Sigl, A security-aware routing implementation for dynamic data protection in zone-based mpsoc, in: Integrated Circuits and Systems Design (SBCCI), 2017 30th Symposium on, IEEE, 2017, pp. 59–64.

[48] T. English, E. Popovici, M. Keller, W. P. Marnane, Network-on-chip interconnect for pairing-based cryptographic ip cores, Journal of Systems Architecture 57 (1) (2011) 95–108.

[49] C.-P. Young, C.-C. Chia, L.-B. Chen, J. Huang, On-chip-network cryptosystem: a high throughput and high security architecture, in: Circuits and Systems, 2008. APCCAS 2008. IEEE Asia Pacific Conference on, IEEE, 2008, pp. 1276–1279.

[50] J. Sepúlveda, D. Flórez, G. Gogniat, Reconfigurable group-wise security architecture for noc-based mpsocs protection, in: Integrated Circuits and Systems Design (SBCCI), 2015 28th Symposium on, IEEE, 2015, pp. 1–6.

[51] J. Sepulveda, D. Flórez, V. Immler, G. Gogniat, G. Sigl, Efficient security zones implementation through hierarchical group key management at noc-based mpsocs, Microprocessors and Microsystems 50 (2017) 164–174.

[52] G. Sharma, R. A. Sahu, V. Kuchta, O. Markowitch, S. Bala, Authenticated group key agreement protocol without pairing, in: International Conference on Information and Communications Security, Springer, 2017, pp. 606–618.

[53] G. Sharma, V. Kuchta, R. A. Sahu, S. Ellinidou, O. Markowitch, J.-M. Dricot, A twofold group key agreement protocol for noc based mpsocs, in: 2018 16th Annual Conference on Privacy, Security and Trust (PST), IEEE, 2018, pp. 1–2.

[54] L. Cong, W. Wen, W. Zhiying, A configurable, programmable and software-defined network on chip, in: Advanced Research and Technology in Industry Applications (WARTIA), 2014 IEEE Workshop on, IEEE, 2014, pp. 813–816.

[55] S. Ellinidou, G. Sharma, S. Kontogiannis, O. Markowitch, J.-M. Dricot, G. Gogniat, Microlet: A new sdnoc-based communication protocol for chiplet-based systems, in: 2019 22nd Euromicro Conference on Digital System Design (DSD), IEEE, 2019, pp. 61–68.

[56] A. Scionti, S. Mazumdar, A. Portero, Towards a scalable software defined network-on-chip for next generation cloud, Sensors 18 (7) (2018) 2330.

[57] R. Sandoval-Arechiga, J. Vazquez-Avila, R. Parra-Michel, J. Flores-Troncoso, S. Ibarra-Delgado, Shifting the network-on-chip paradigm towards a software defined network architecture, in: Computational Science and Computational Intelligence (CSCI), 2015 International Conference on, IEEE, 2015, pp. 869–870.

[58] R. Sandoval-Arechiga, R. Parra-Michel, J. Vazquez-Avila, J. Flores-Troncoso, S. Ibarra-Delgado, Software defined networks-on-chip for multi/many-core systems: A performance evaluation, in: Proceedings of the 2016 Symposium on Architectures for Networking and Communications Systems, ACM, 2016, pp. 129–130.

[59] A. Scionti, S. Mazumdar, A. Portero, Software defined network-on-chip for scalable cmps, in: High Performance Computing & Simulation (HPCS), 2016 International Conference on, IEEE, 2016, pp. 112–115.

[60] K. Berestizshevsky, G. Even, Y. Fais, J. Ostrometzky, Sdnoc: Software defined network on a chip, Microprocessors and Microsystems 50 (2017) 138–153.

[61] A. Fathi, K. Kia, A centralized controller as an approach in designing noc, International Journal of Modern Education and Computer Science 9 (1) (2017) 60.

[62] M. Ruaro, H. M. Medina, F. G. Moraes, Sdn-based circuit-switching for many-cores, in: 2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), IEEE, 2017, pp. 385–390.

[63] M. Ruaro, H. M. Medina, A. M. Amory, F. G. Moraes, Software-defined networking architecture for noc-based many-cores, in: Circuits and Systems (ISCAS), 2018 IEEE International Symposium on, IEEE, 2018, pp. 1–5.

[64] ARM, Security technology building a secure system using trustzone technology (white paper), ARM Limited (2009).

[65] V. Costan, S. Devadas, Intel sgx explained., IACR Cryptology ePrint Archive 2016 (086) (2016) 1–118.

[66] L. Sanders, Isolation methods in zynq ultrascale+ mpsocs (2017).
URL https://www.xilinx.com/support/documentation/application_notes/xapp1320-isolation-methods.pdf

[67] C. Bao, A. Srivastava, 3d integration: New opportunities in defense against cache-timing side-channel attacks, in: 2015 33rd IEEE International Conference on Computer Design (ICCD), 2015, pp. 273–280. doi:10.1109/ICCD.2015.7357114.

[68] J. Dofe, P. Gu, D. Stow, Q. Yu, E. Kursun, Y. Xie, Security threats and countermeasures in three-dimensional integrated circuits, in: Proceedings of the on Great Lakes Symposium on VLSI 2017, GLSVLSI '17, ACM, New York, NY, USA, 2017, pp. 321–326. doi:10.1145/3060403.3060500.
URL http://doi.acm.org/10.1145/3060403.3060500

[69] J. Dofe, Q. Yu, H. Wang, E. Salman, Hardware security threats and potential countermeasures in emerging 3d ics, in: 2016 International

Great Lakes Symposium on VLSI (GLSVLSI), 2016, pp. 69–74. doi:10.1145/2902961.2903014.

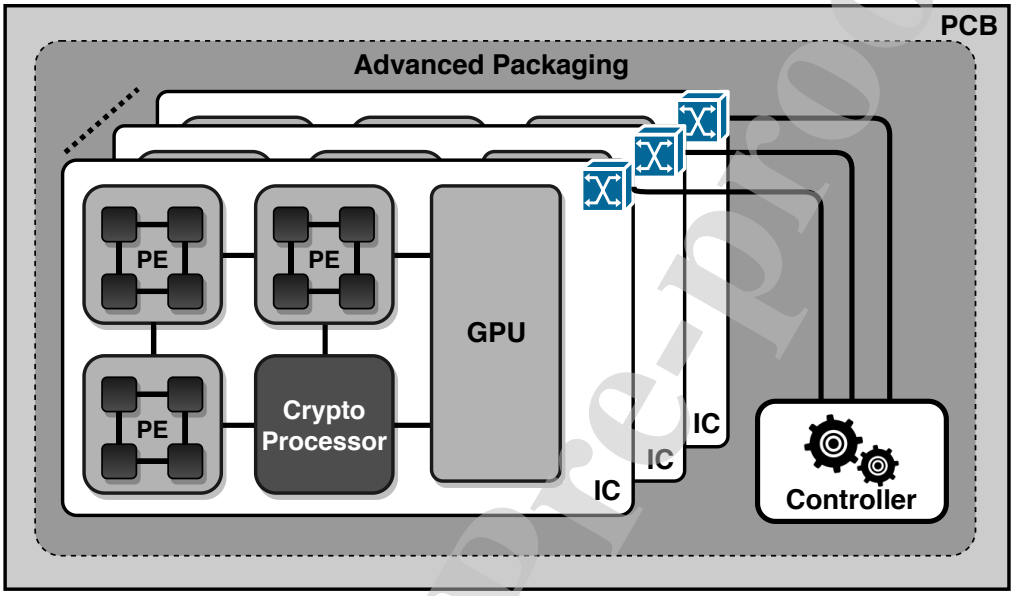


Figure 1: Cloud-of-Chips Platform Architecture

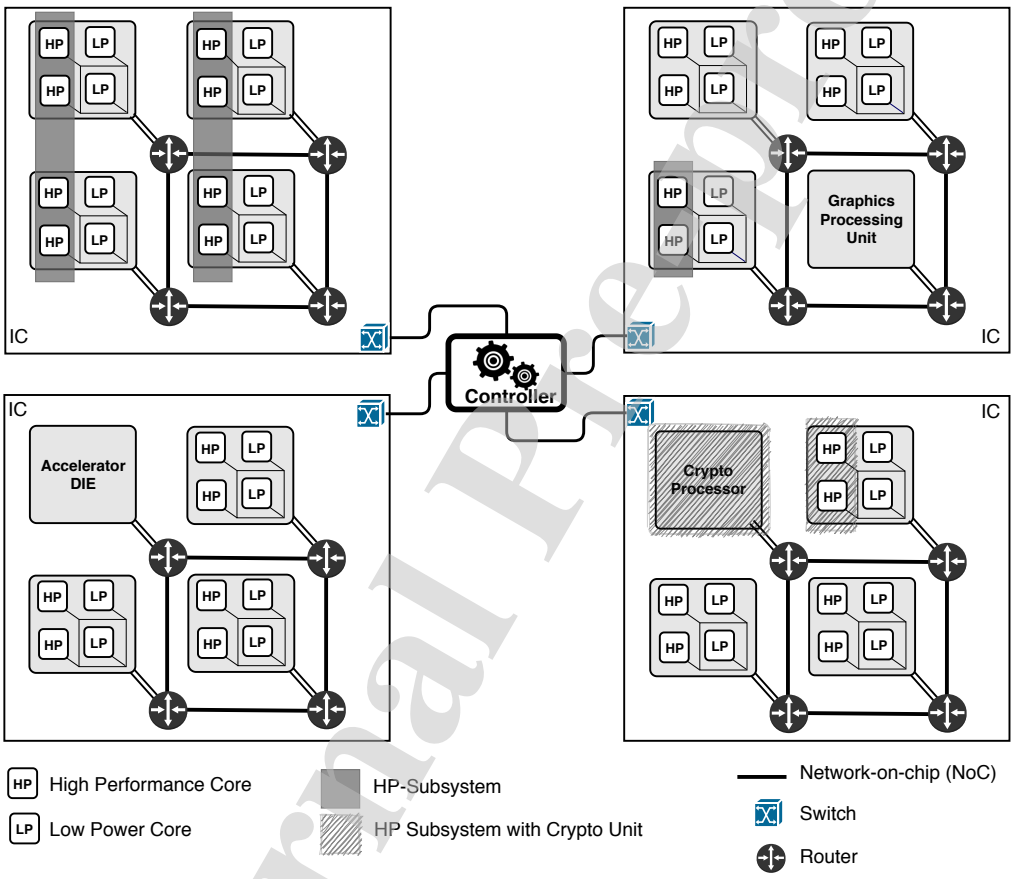


Figure 2: Subsystems on CoC Platform

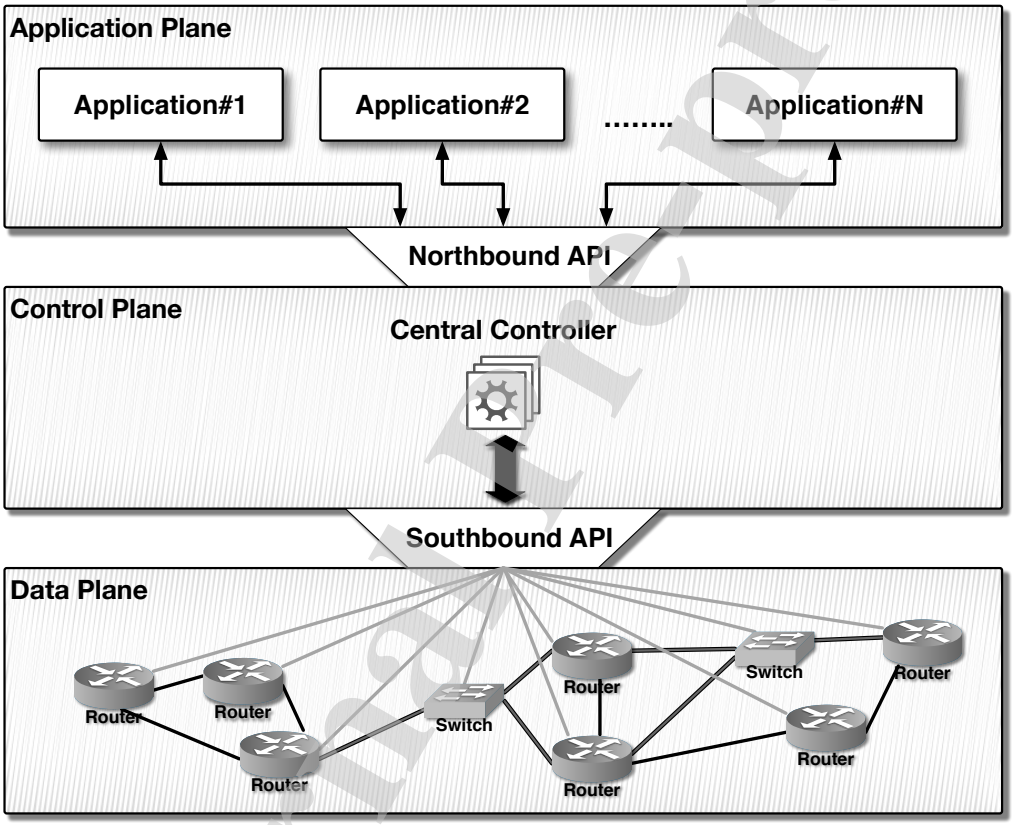


Figure 3: SDN Architecture

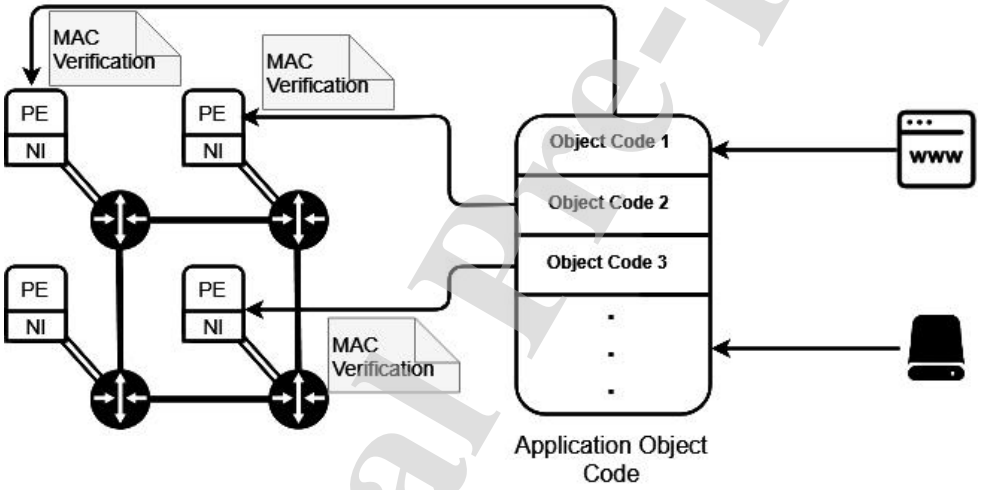


Figure 4: Application Admission

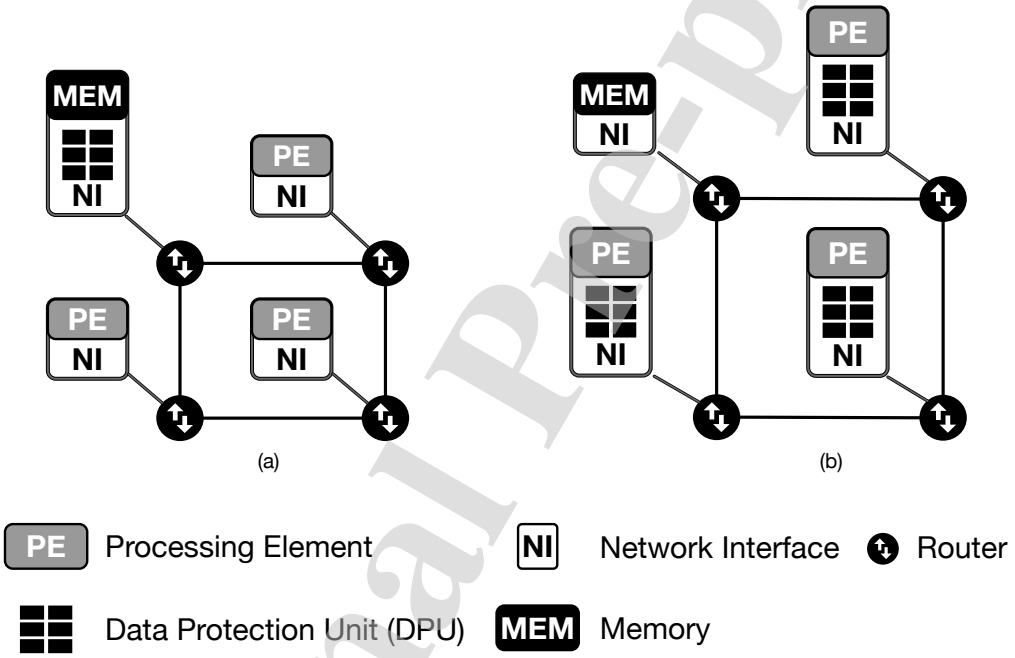


Figure 5: Secure Memory Access (a) DPU on Memory end (b) DPU in individual PE

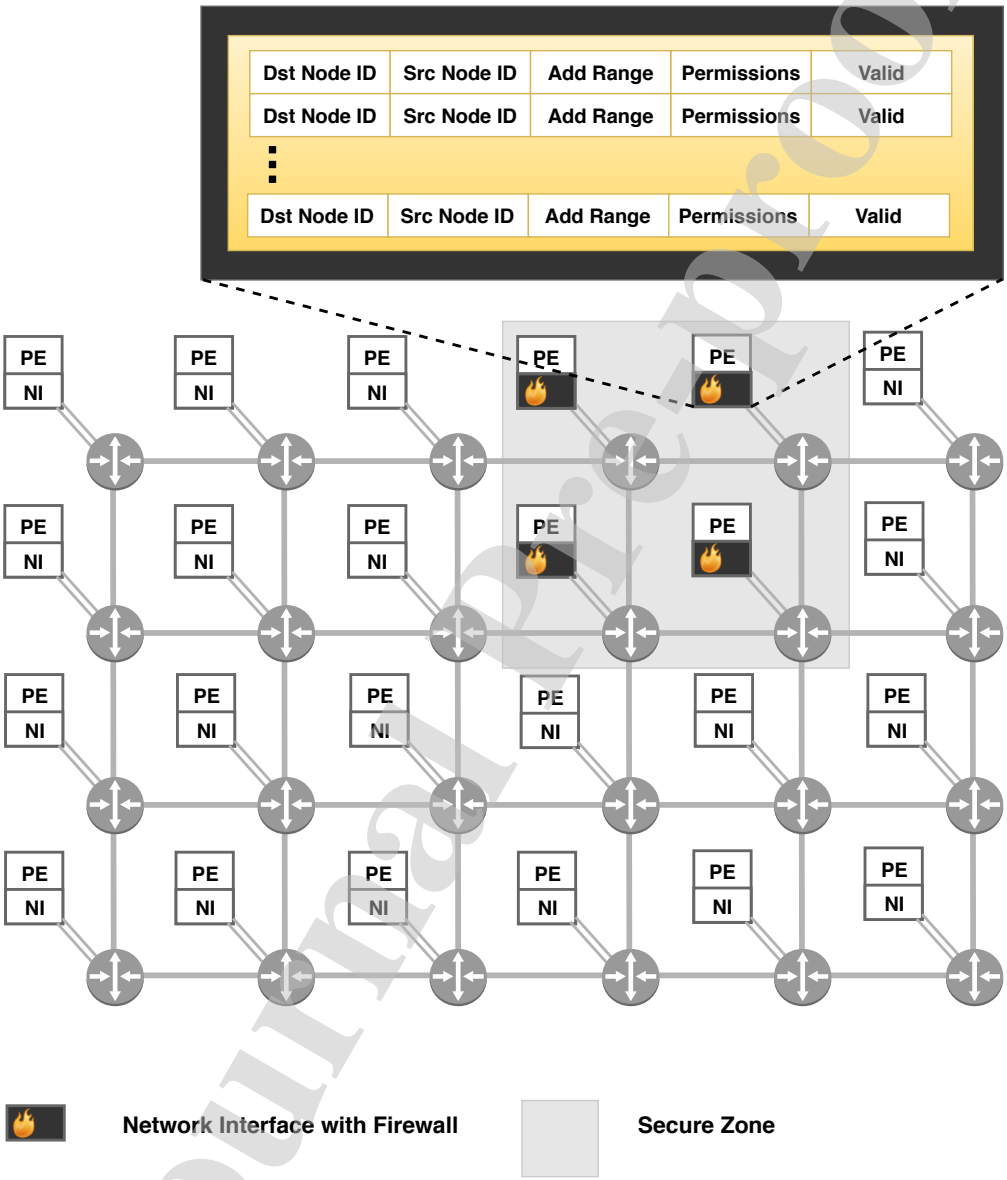


Figure 6: MPSoC Firewall Protection

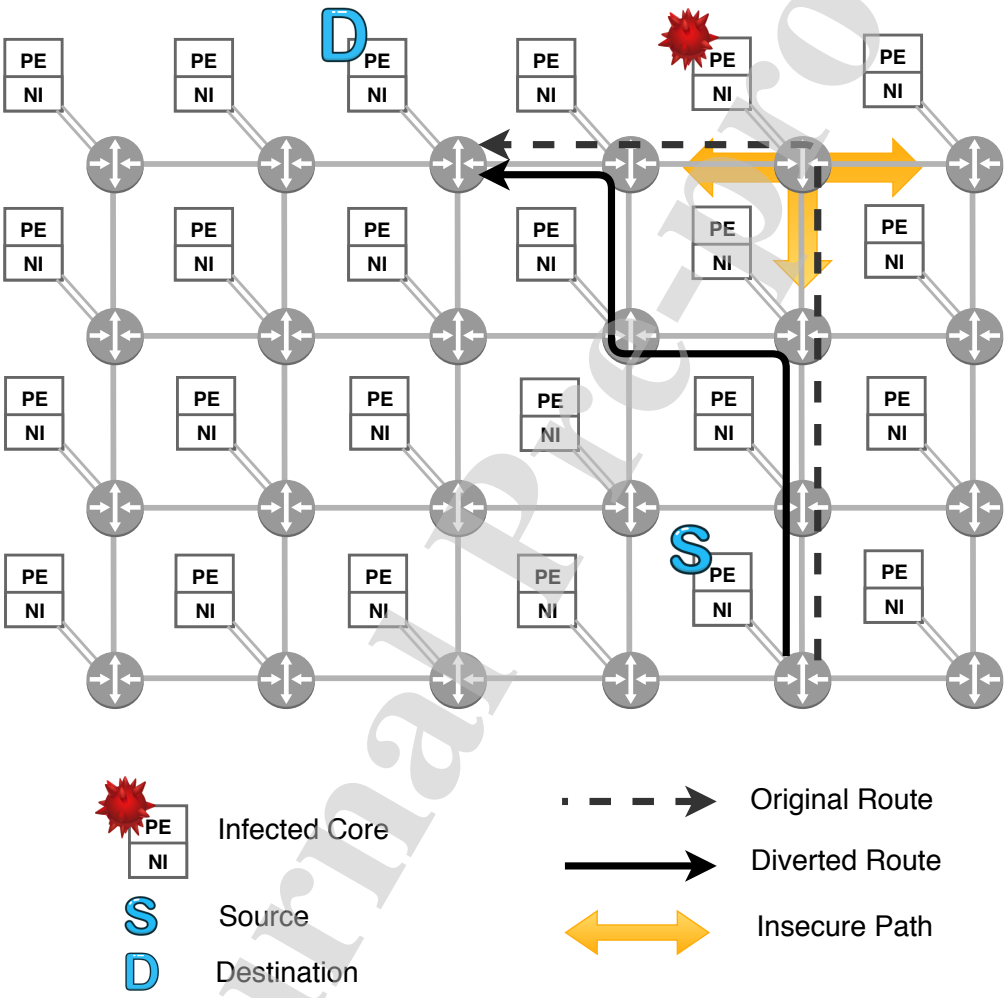


Figure 7: Change of route to avoid infected IP

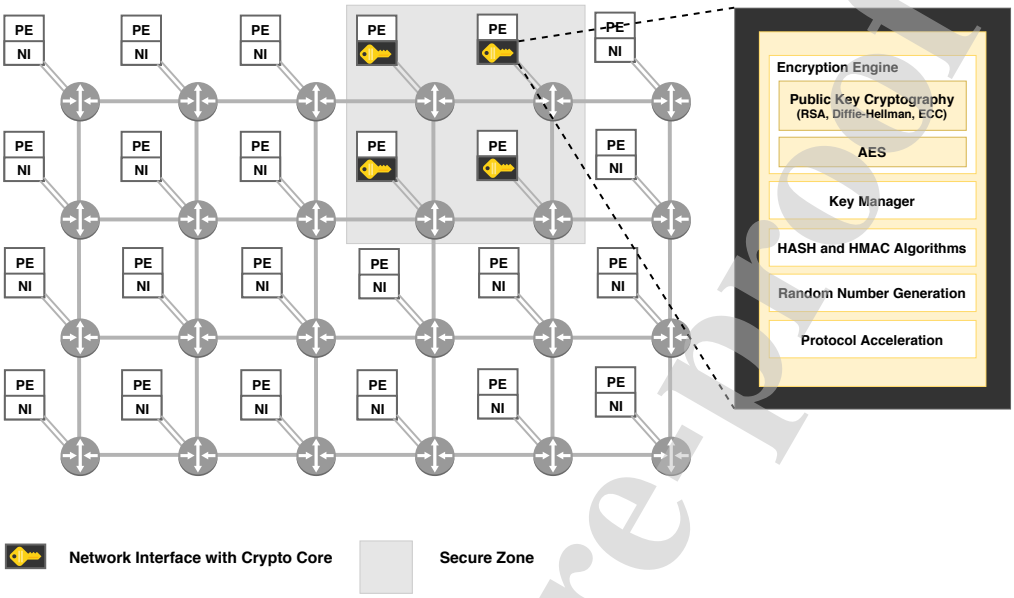


Figure 8: MPSoC with Crypto Cores

Table 1: MPSoC Threats and Countermeasures

Threat	**Countermeasure**	**Reference**
Secure Application Admission	Verify MAC	[32]
Secure Memory Access		
– Secret Information Extraction	Legitimate access to memory blocks	[34][35]
– Hijacking	Legitimate access to memory blocks	[34][35]
Secure Running Environment		
– Hardware Trojan	Secure Supply Chain	[36]
– Timing Attack	Masking Technique, Power Consumption Randomization	[36]
– DoS	Exclusive Resources for Computation and Communication	
	– Firewall/Wrapper	[43],[44],[45]
	– Security-Aware Routing	[42],[47]
	– Key Agreement	[45],[49],[50],[53]

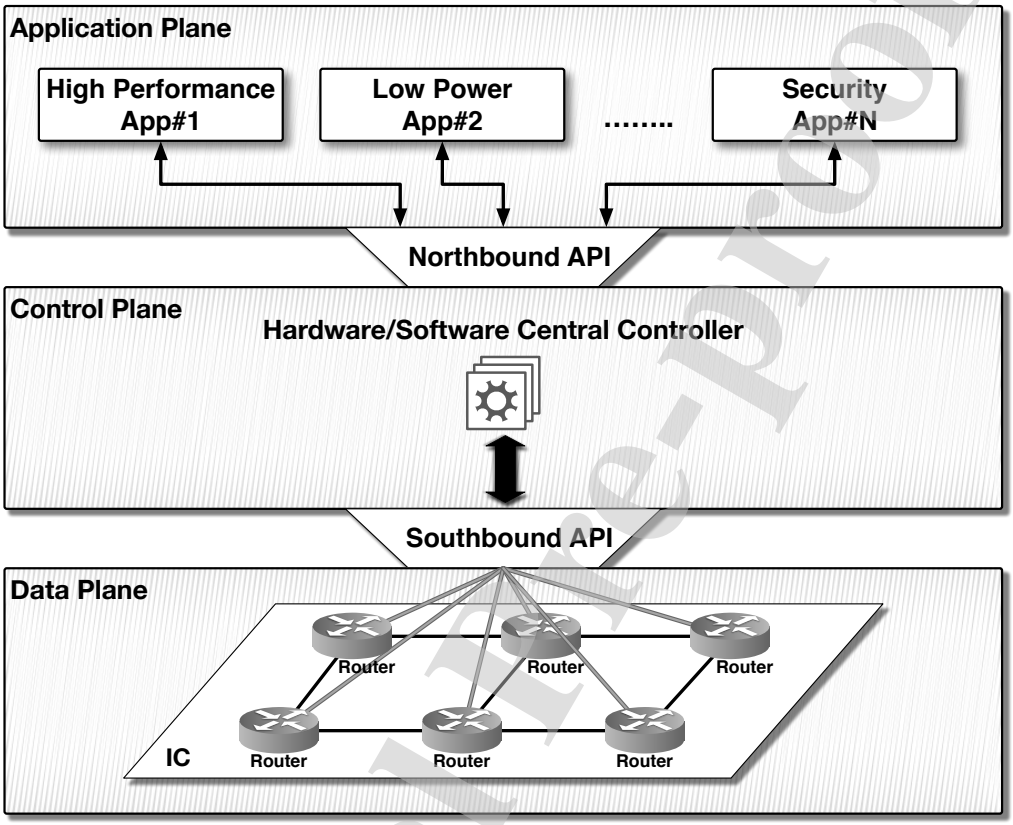


Figure 9: SDNoC Architecture

Table 2: Secure Zones at run-time

Approach	Secure Zone
Firewall	Continuous
Wrapper	Continuous
Firewall+Secure Routing	Continuous & Disrupted
Key Agreement	Continuous & Disrupted
SDNoC	Continuous & Disrupted

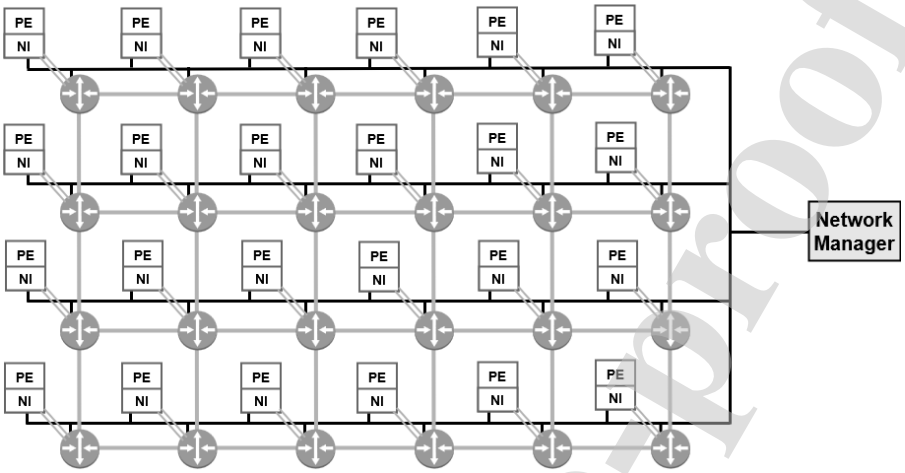


Figure 10: SDNoC [60]

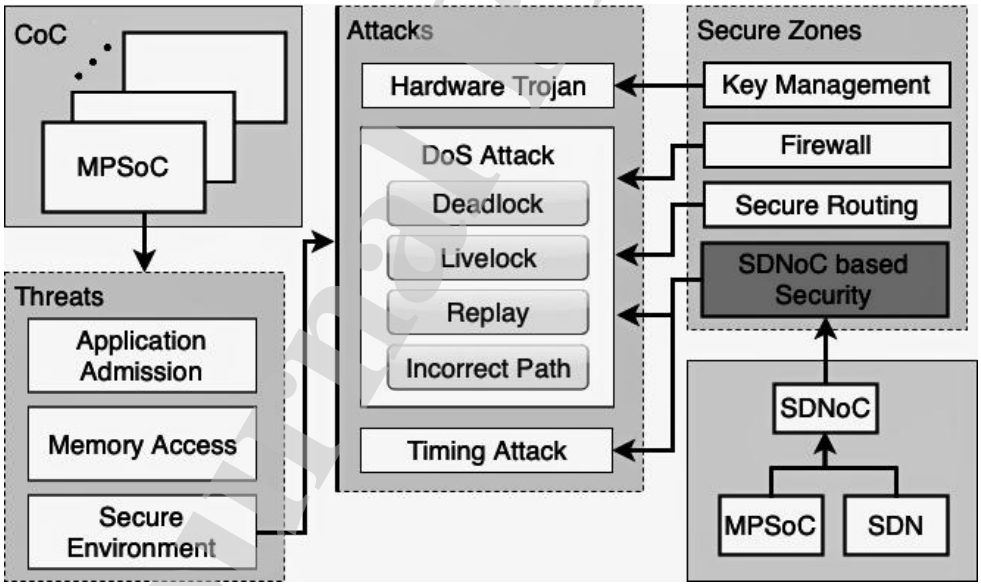


Figure 11: SDNoC as a Security Alternative

Table 3: Comparison of Existing Approaches with Security Features

Approach	Replay	Incorrect Path	Deadlock	Livelock	Cache-Timing	Hardware Trojan
Firewall	√	√	√	√	×	×
Wrapper	√	√	√	√	×	×
Firewall + Adaptive Routing	√	√	√	√	×	×
Key Agreement	×	×	×	×	×	√
SDNoC	√	√	√	√	√	×
SDNoC + Key Agreement	√	√	√	√	√	√

Gaurav Sharma is currently working as a researcher at the Université libre de Bruxelles, Belgium. He received his Ph.D and M.E degree in Computer Science & Engineering from Thapar University, Patiala, India. His current research is based on exploring security threats in Multi-Processor System-on-chips (MPSoCs). He also worked with routing and security issues in Ad hoc networks. Dr. Sharma is a member of IEEE since 2008 and he has authored/coauthored more than 45 journal/conference articles and book chapters. He serves as reviewer of IEEE Systems Journal, IEEE Sensors Journal, Future Generation Computer Systems and Journal of Information Security and Applications. He was the TPC member of GlobeCom'18, IndiaCom'18, CRIS'18, ICCCA'17 and SPIN'16. Dr. Sharma also edited a special issue on "Advanced Research in Privacy and Forensic Analytic of Web Engineering", International Journal of Information Technology and Web Engineering (IJITWE), IGI Global.

Jean-Michel Dricot leads research on network security with a specific focus on the IoT (Internet of Things) and wireless networks. He teaches communication networks, mobile networks, internet of things, and network security. Prior to his tenure at the ULB, Jean-Michel Dricot obtained a PhD in network engineering, with a focus on wireless sensor networks protocols and architectures. After his PhD, he joined France Telecom R&D (Orange Labs) in Grenoble, France, as a research engineer. He started there a project aiming at securing lightweight communication protocols, with a specific focus on wireless smart meters and body area networks. Next, he moved back to the ULB, in the machine learning group, where he worked on IoT-based localisation techniques. In 2010, Jean-Michel Dricot was appointed professor at the Université Libre de Bruxelles, with a tenure in mobile and wireless networks. He is author or co-author of more than 100+ papers published in peer-reviewed international Journals and Conferences and served as a reviewer for European projects.

Dragomir Milojevic received his Masters and PhD degrees in Electrical Engineering from the Ecole polytechnique de Bruxelles, at Université libre de Bruxelles (ULB), Belgium. He is a professor of digital electronics and digital systems design at ULB where he co-founded and co-directs Parallel Architectures for Real-Time Systems (PARTS) research group. In 2004 he joined IMEC where he first worked on multi-processor and Network-on-Chip architectures for low-power multimedia systems. Since 2008 he is working on design-technology co-optimization of advanced technology nodes and design methodologies and tools for technology aware design of 3D integrated circuits. Dragomir Milojevic authored or co-authored more than 100 journal and conference articles, and served as technical program committee member to several conferences in the field.

Soultana Ellinidou is a PhD researcher at Université Libre de Bruxelles, Brussels, Belgium. She holds a bachelor and master degree in Engineering Informatics & Telecommunications from University of Western Macedonia, Kozani, Greece. She is currently obtaining a PhD degree in Engineering and Technology from Université Libre de Bruxelles. The title of her PhD is "Protocols and algorithms for secure Software Defined Network-on-Chip (SDNoC)". Her research interests cover the area of Internet of Things (IoT), Wireless Sensor Network (WSN), cyber security, Software Defined Network (SDN), System on Chip (SoC), Network-on-Chip (NoC), and Hardware Trojans (HT).

Olivier Markowitch received his Masters and PhD degrees in Computer Sciences from the Faculty of Sciences at Université libre de Bruxelles (ULB), Belgium. He is a professor of computer sciences and cryptography at ULB where he co-founded the Cybersecurity Research Center as well as the Belgian Inter-Universities Master in Cybersecurity. He is working on the design and analysis of secure protocols, cryptography protocols and secure communications. Olivier Markowitch authored or co-authored more than 100 journal and conference articles, and served as technical program committee member of several conferences.

Georgios Bousdras received his diploma in Electrical and Computer Engineering at University of Thessaly, Greece. He is a PhD student at Université libre de Bruxells, in Belgium on the area of Embedded Electronics. His PhD thesis addresses the topic of "Template architectures for highly scalable, many-core Heterogeneous SoC". His area of interest is Multiprocessor System-on-Chip (MPSoC), Network-on-Chip (NoC), 3D circuit integration (IC), and image processing on reconfigurable hardware systems.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☒The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Gaurav Sharma
Georgios Bousdras
Soultana Ellinidou
Olivier Markowitch
Jean-Michel Dricot
Dragomir Milojevic