Understanding MPSoCs: Exploiting Memory Microarchitectural Vulnerabilities of High Performance NoC-Based MPSoCs

Johanna Sepulveda
Technische Universität München
Munich, Germany
johanna.sepulveda@tum.de

Cezar Reinbrecht
Delft University of Technology
Delft, Netherlands
C.R.Reinbrecht@tudelft.nl

Siavoosh Payandeh Azad,
Behrad Niazmand, Gert Jervan
Tallinn University of Technology
Tallinn, Estonia
siavoosh.azad,behrad.niazmand,
gert.jervan@ttu.ee

ABSTRACT

Multi-Processor Systems-on-Chips (MPSoCs) are the key enabler technology for current and future applications. However, the high on-chip connectivity, the programmability and IPs reusability, also introduce security concerns. Problems arise when applications with different trust and security levels share the MPSoC resources. One of the potent threats that MPSoCs see themselves exposed to are the so-called side-channel attacks (SCA). In this work, we explore the cache-based side-channel attacks optimized by the communication structure. We evaluate the vulnerability of the different NoC-based MPSoC memory configuration against micro-architectural side channel attacks. Our attack targets an MPSoC AES T-Table implementation. We explore the impact of the MPSoC organization on the NoC timing attack. We present the huge impact on the memory organization and present two attack metrics: efficacy and efficiency. Our results show that NoC-based MPSoCs are vulnerable and that deep memory hierarchies favor the security of the system.

1 INTRODUCTION

In recent years, Multi-Processor System-on-Chips (MPSoCs) have become the key enabler technology for many key industries. These systems integrate a set of different IP-Cores (such as processors, accelerators, peripherals etc.) into one heterogeneous system. These MPSoCs are used for storing and processing sensitive data in many critical applications. However, with the emerging of hyperconnected semiconductor systems in the age of Internet of Things (IoT), these MPSoCs are not working in isolation anymore. During the lifetime of the system, the applications running on the MPSoC are subject to change and further development and it is essential for the system to provide software update support. In order to facilitate such requirement, the MPSoC should be accessed remotely. However, the remote access privileges open the MPSoCs as a target to many security attacks.

Side channel attacks exploit the physical behavior of the MPSoC under operation, such as heat dissipation, power consumption, electromagnetic emanation and timing in order to leak sensitive information. However, many of such attacks require physical access to the MPSoCs which in many cases might not be possible. This turns attractive the attacks able to be executed remotely. Timing attacks exploit the micro-architecture of the MPSoCS such as memory organization, processor configuration and communication infrastructure. Meltdown [9] and Spectre [8] attacks are recent examples of such micro-architectural attacks on MPSoCs.

Previous works have shown that memory-based attacks can be very effective [3]. Furthermore, the communication infrastructure in the MPSoC can be exploited to optimize memory-based attacks [10, 12, 14]. However, in the previous works only the shared caches have been exploited where in modern MPSoCs, different memory organizations are available. In this work, we explore for the first time three memory different organizations: Central, shared and distributed. We evaluate the efficiency of the attack and the impact of the attacker's location in the MPSoC.

The remainder of this paper is organized as follows: Section 2 presents the previous works. The description of the MPSoC and the memory organizations are presented in sections 3 and 4. Section 5 describes the threat model considered in this work and Section 6 provides the experimental results, comparison and discussion. Finally section 7 concludes the paper.

2 RELATED WORK

In a System-on-Chip (SoC) which uses NoC as its communication infrastructure, the network can be taken advantage by an adversary to compromise the overall system security. One type of attacks on NoCs already addressed in the literature is side-channel attacks (SCA). The goal of the attack is to acquire information regarding the sensitive data transmitted over the network via taking advantage of the power consumption, timing information, or electromagnetic leaks. The focus of this paper is on timing SCA in MPSoCs which use a shared memory over the network (more specifically cache timing attacks).

Previous research has shown that NoC timing attacks are practical. In [15], two mechanisms *i.e.* random arbitration and adaptive routing are proposed in order to avoid SCA raising from resource sharing in NoCs. By isolating the sensitive communication, the network is protected against time-driven attacks. Moreover, by dynamically allocating communication resources, the system is prevented from possible Denial of Service (DoS) attacks and the mechanisms do not degrade system performance (unlike approaches such as [17]). Authors of [10] have introduced a NoC-based cache timing attack, named Earthquake. This work explores the vulnerabilities of the differential cache-collision attacks, proposed previously in [3]. The proposed attack in [10] allows to measure only the first three rounds of Advanced Encryption Standard (AES), improving the detection of wide-collisions. Compared to [3], the work in [10]

SAMOS XVIII, July 15–19, 2018, Pythagorion, Samos Island, Greece
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-6494-2/18/07…$15.00
https://doi.org/10.1145/3229631.3239367

reduces the number of measurements and thus, the overall attack time.

In [12], a timing side-channel attack, named Prime+Probe for NoC-based systems is introduced. Despite reducing the search space by using 80 encryptions, the practical execution of the attack has limitations regarding synchronization and accuracy. In order to mitigate the effects of such attack, a secure platform, Gossip NoC [11] is used to evaluate the improved version of the attack. Authors of of [14] have investigated cache attacks on SoCs implementing bus-based communication. In [7], a methodology is proposed based on probabilistic information flow graph (PIFG), which models the interaction between the victim program, the attacker program and the cache architecture. Moreover, a metric named Probability of Attack Success (PAS) is introduced. By means of these, the security of different proposed secure cache architectures are verified and their resilience to cache side-channel attacks is also evaluated. Authors of [13] have proposed a secure enhanced NoC router architecture, which is capable of dynamically configuring the router memory space according to the communication and security properties of the traffic. With the proposed approach, they avoid timing attacks by turning the attacker oblivious of the sensitive traffic.

The previous mentioned works have explored timing side-channel attacks (including cache-based attacks in some works) in NoC-based MPSoCs. However, they have not yet explored the vulnerability of the different NoC-based MPSoC memory configuration for micro-architectural side channel attacks. The experimental results of this work will show that NoC-based MPSoCs are vulnerable and we propose countermeasures to avoid such a threat.

3 MPSOC

MPSoCs are a complete computational system integrating multiple processing elements on the same integrated circuit [**?**]. They are organized in tiles, composed of processors, co-processors, hardware accelerators, memories and other Intellectual Property (IP) cores. Such tiles are interconnected through a Network-on-Chip (NoC), composed by routers and links, which allows the communication exchange among the different MPSoC components. Designing an MPsoC demands the selection of several parameters. MPSoCs are characterized by four models: i) Computational architecture, which depends on the organization of the processing structure; ii) Memory, including the structure to store information; iii) Communication, which depends on the organization of the structure that performs the data exchange; and iv) Software, including the programs required to perform the different functionalities of the MPSoC. Memory organization has a strong impact on the MPSoC performance. The focus of this work is to study the impact of the memory organization on the microarchitectural NoC timing attacks. This work uses Bonfire Network on Chip framework [1]. The targeted NoC has a 2D mesh topology where each tile of the network consists of a wormhole switching router. The communication between routers is handled using credit-based flow control. Each input channel of the router consist of an input FIFO buffer and a routing unit, implemented using logic-based distributed routing mechanism [6] (removing the need for any routing tables). The output channels are allocated using round-robin. The Bonfire router does not contain any output buffers. Fig. 1 depicts the block diagram of the Bonfire router.

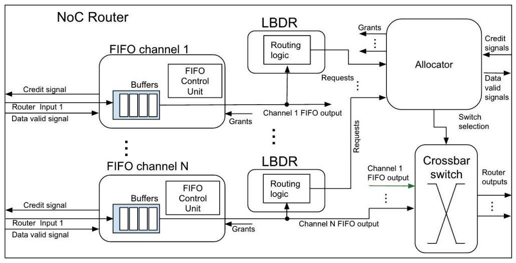

Figure 1: block diagram of Bonfire router

The introduction of SoCs in IoT environments promotes the integration of security primitives such as the Advanced Encryption Standard (AES). AES is a symmetric key encryption algorithm, which is able to encrypt 128 bits of data with key lengths of 128 bits using 10 rounds. The algorithm includes a set of intermediate states which are operated through 10 rounds. Each state is composed by the following operations: AddRoundKey (XORing the state with the current round key), SubByte (byte substitution), ShiftRow (byte transposition) and MixColumn (matrix multiplication). In order to improve the performance of AES implementations, the SubByte, ShiftRow and MixColumn operations operations can be performed using tables. Such tables which include pre-computed values are called Transformation tables (T-tables) [5]. In such case, these operations are reduced to four look-up tables (*i.e.* T0 - T3) whose entries are simply XORed. These tables must be stored using the memory architecture of the MPSoC.

4 MPSOC MEMORY ORGANIZATION

MPSoC architectures are based on memory hierarchies, consisting in the integration of several levels of cache (L1, L2, L3) and DRAM. Processors and L1 are sometimes integrated into a single tile. L1 stores the data and instruction data of an application. When a cache miss occurs on L1, the cache coherency mechanism initiates an access to the shared L2 cache, located, usually, on another distant core. The memory hierarchy can be categorized as i) main memory, where all the data is stored in a big and usually external memory; ii) shared memory, where a single memory stores the data of different processors, which may or may not share common memory spaces. Accesses are managed by the operating system. However, in the presence of multiple access requests, the shared nature of this type of memory creates a bottleneck that degrades the performance of the applications; iii) distributed memory, where several memories are distributed along the MPSoC's tiles. Each memory is exclusive for each processing element, thus avoiding address space sharing. Despite providing data isolation into the MPSoC, this technique demands intense communication.

5 THREAT MODEL

To launch a successful timing attack in the NoC, the attacker requires environmental information. Additionally, the victim's system should allow actions which, in turn, will enable the malicious software to perform the attack.

5.1 Requirements for the attack

More specifically, the conditions of a successful timing attack in the NoC can be listed as:

(1) System Susceptibility: The attacker is able to run malicious software inside the MPSoC.
(2) Attacker's Reach: The attacker can directly or indirectly communicate with the victims.
(3) Sharing the Sensitive Path: The attacker's observable path in the network has an intersection with the sensitive path.

System Susceptibility: This condition highlights the vulnerability of the target MPSoC. The attacker can use different techniques to tamper with the software and to infect an IP. As an example, the attacker can use malicious software (malware) in order to perform read/write operations in the restricted memory addresses. This, in turn, may result in a change in the victim IP's behaviour, resulting in an infected IP. Moreover, software weaknesses such as buffer overflows [4] and other similar techniques can be exploited for such a purpose.

Attacker's Reach: This condition highlights the attacker's ability to communicate with the attacker's primary target, for example via the victim's processor and the shared cache. Attacker's first task after a successful code infection, would be to identify the logical addresses of the elements. Typical MPSoCs provide system functions which can ask jobs form other IPs. With the main objective of such interaction being triggering the victim to perform sensitive operations (*e.g.* encryption). The attacker has three possibilities to acquire the logical addresses:

(1) the logical addresses are provided in MPSoC's datasheet;
(2) functions from an API perform the communication with the target IPs; or
(3) functions from an API ask the service to a centralized manager, and it passes to the destination IP. This scenario considers a highly secure system, where the trusted IPs are entirely isolated.

Sharing the Sensitive Path: Once the attacker has the ability to trigger a sensitive operation in the victim, to execute a timing attack, it is crucial for the attacker to observe the victim node's traffic. To enable such observation, it is essential that the malicious software shares sections of the sensitive path. This can be achieved by the attacker asking the crypto-processor (primary target) to perform an encryption while asking another service from a different IP-core. Next, the attacker will test if the established communication traffic intersects with the sensitive path systematically. If the attacker can not establish a communication path that satisfies this requirement, a different IP core must be infected. Due to specific cache access pattern of AES, it is practical for the attacker to perform such a search for sensitive traffic.

This process can be further optimized if the MPSoC provides more implementation details, such as topology and routing algorithm used in NoC.

5.2 Description of the timing attack

The fundamental mechanism exploited by the attack is that the AES T-tables are accessed depending on the secret key. The goal of the attacker is to detect whether T-table entries stored on memory lines have been evicted or not. The attacker should use legitimate operations (read and write) and exploit the fact that the time to retrieve information varies according to the amount of memory misses and hits. Detecting the optimal point of attack may reduce the attack effort (measured in terms of the time spent to retrieve the secret key). The optimal point to perform the attack is after the first round of the AES encryption [14]. In order to detect the first round, the communication structure can be used. By monitoring the shared NoC, it is possible for the attacker to detect, in a non-intrusive way, the optimal point in time to probe the memory, as described in [10]. An attacker, which started an AES encryption, is then injecting packets to the NoC. In order to perform the AES encryption, the T-tables should be accessed, that result in the injection of a big packet (integrating several flits) from the memory where the T-tables are stored, to the AES IP core. As the communication structure is shared, the injection of the T-table packet causes a contention of the attacker communication. The degradation of the injection throughput of the attacker alerts the possible T-Table retrieval. As a consequence, the attacker can now start a memory attack at the optimal point of time, thus allowing the faster secret key retrieval, when compared to a memory attack.

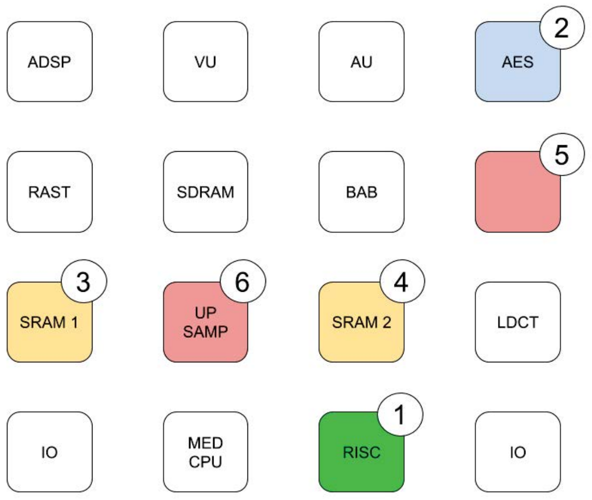

Figure 2: Mapping of MPEG-4 and AES Applications at the target MPSoC.

6 EXPERIMENTAL WORK

The goal of the experiments is to evaluate the impact on the efficacy and the efficiency of the NoC Timing Attack over different memory hierarchy configurations. The retrieval of the secret key of a AES T-Table implementation was the target of attack. The efficacy of the attack is defined as the amount of sensitive data observed by an attacker. The efficiency of the attack measures the amount of correctly guessed packets during an attack (also referred as true positives). According to the storage mechanism used to save the T-Tables, three memory organization were used in the experimental setup: i) T-Tables of AES at main memory; ii) T-Tables of AES at a shared cache L2; and iii) T-Tables of AES at a distributed shared cache L2.

The MPSoC platform used to conduct the experimental work integrates 12 IP cores which exchange data through a 4x4 mesh-based NoC. The MPSoC platform is shown in Fig. 2. The communication structure of the MPSoC is the NoC Bonfire [1]. It uses a XY routing algorithm and router which is able to store 8 flits at each input buffer, where each flit is defined as 32 bits.

Usually the AES encryption is not executed in the MPSoC as a stand-alone application. Multiple traffic is simultaneously generated

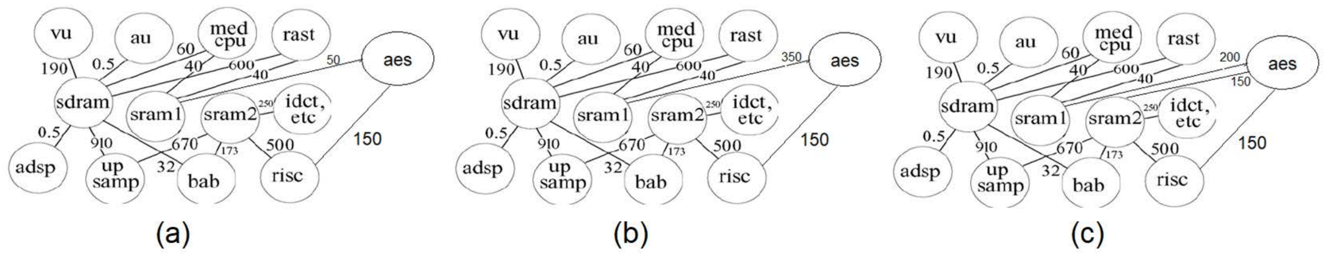

Figure 3: Communication graphs of each experiment. (a) Main Memory, (b) Shared Cache, and (c) Distributed Shared Cache.

for different applications which is executed concurrently. This fact produces a increase of the noise in the attacker's measurements. In order to emulate this effect, we include, simultaneously with the AES execution, the MPEG-4. This application was chosen due its high computation and communication requirements, which include high level of parallelism, high number of messages exchanged and high number of memory requests. The MPEG-4 communication graph and mapping for a 4x4 MPSoC are presented in [16]. The communication and mapping graphs of the concurrent execution of the MPEG-4 and AES applications are shown in Fig. 3 and Fig. 2, respectively. In order to speedup the simulation time, traffic generators were used to emulate the communication behavior.

Fig. 2 highlights the most important IP cores for the experimental setup. The IP core (1), the RISC processor, is the master processor responsible to trigger and control the MPEG-4 and AES applications. The IP core (2) is the cryptographic co-processor used to execute the AES encryption/decryption. In order to complete the AES operation, T-Table accesses should be done. IP core (3) and (4) are used to store the T-Tables. Two potential locations (5) and (6) were chosen as the possible attacker locations. An attacker located at IP core (5) is able to observe the traffic close to the AES IP. On the other hand, an attacker located at (6) is able to observe the traffic closer to the memories. Both locations share the sensitive path (defined in the threat model).

In order to perform the NoC Timing Attack, an attacker infects an IP core of the MPSoC (5 or 6) and starts injecting packets in the network at higher data rates. Then, the attacker measures the injection latency of its packets (time require to inject a packet) in order to understand the network behavior. As a result, a set of traces are captured and divided in different thresholds, as shown in Fig. 4. These threshold are variable and define the boundaries in which sensitive data can be identified. The refinement of the boundaries also incurs in the enhance of the success of the attack, as shown in [12]. Since this task is not trivial, our experiments set different threshold levels, and put in the final results to understand the impact of this feature in the efficacy and efficiency of the attack.

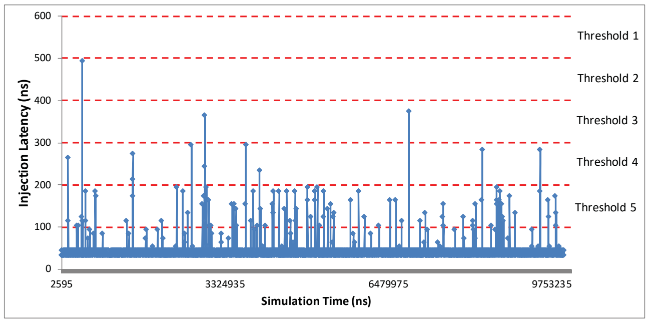

Figure 4: Injection latency (ns) vs. Simulation time (ns)

Four parameters were varied in order to evaluate the impact of the MPSoC organization in the attack: i) memory organization; ii) variation of the network interface buffer sizes (8, 16,32 and 64 flits); iii) location of the attacker (6 and 5); and iv) threshold (100, 150, 200 and 300). Results are presented in Table 1. The experiments are described as:

- EXP1: T-Tables of AES at Main Memory - Transmission of memory blocks of 256 words;
- EXP2: T-Tables of AES at Shared Cache L2 - Transmission of cache lines of 32 words; and
- EXP3: T-Tables of AES at Distributed Shared Cache L2 - Transmission of cache lines of 32 words.

According to the results presented at Table 1, the attacker at location (6) obtained very good results in the memory configuration of EXP 1. As the main memory sends blocks of data, the packets that are injected into the NoC include several flits. As a consequence, the latency degradation of the attacker is easily observable. Results show that for the EXP1 configuration, the higher the threshold, the higher attack efficiency is achieved. However, for the threshold equal to 300, where only peaks are check, some sensitive information can be lost. The attacker was able to observe only 66% of the packets. For the EXP 1, the threshold equal to 200 presents the best results, being able to observe 70% of all sensitive packets (efficacy) with a successful rate (efficiency) of 94%. According to the results, we can conclude that store the T-Table in the main memory, without including a memory hierarchy, increases considerable the vulnerability of the system to NoC-timing attacks.

The results of EXP2 and EXP3 presented lower values of efficacy and efficiency compared to EXP1. Both use cache memories to store the T-Tables, which inject smaller packets into the NoC. Hence, small sensitive packets can be better camouflaged with the normal NoC traffic, thus turning difficult its detection by the attacker. In these scenarios, the attacker must select between improving the efficacy or efficiency of the attack. If the attack requires a higher amount of sensitive packet detection, the efficacy should be prioritized and the lower thresholds should be used. However, if the attack requires high quality of detection, the efficiency should be prioritized, which represents the selection of higher thresholds. For typical timing attacks, such as the attack proposed by Bernstein [2], the quality is more important. Consequently, for EXP2 and EXP3, where few sensitive data is detected, the time to collect sufficient data to perform the attack increases.

Table 1: Experimental results for percentage of false positives and efficiency for attacks at locations 11 and 5 under different threshold and buffer size values

Threshold	Buffer Size	Attack Location 06						Attack Location 05					
		EXP 1		EXP 2		EXP 3		EXP 1		EXP 2		EXP 3	
		Efficacy (%)	Efficiency (%)	Efficacy (%)	Efficiency (%)	Efficacy (%)	Efficiency (%)	Efficacy (%)	Efficiency (%)	Efficacy (%)	Efficiency (%)	Efficacy (%)	Efficiency (%)
100	8	75%	30%	12%	41%	11%	36%	62%	11%	19%	24%	21%	32%
	16	73%	29%	12%	41%	11%	37%	62%	11%	19%	24%	22%	32%
	32	73%	29%	12%	41%	11%	37%	62%	11%	19%	24%	22%	32%
	64	73%	29%	12%	41%	11%	37%	62%	11%	19%	24%	22%	32%
150	8	73%	49%	6%	48%	6%	37%	57%	17%	12%	27%	13%	35%
	16	70%	48%	7%	49%	6%	36%	57%	18%	12%	27%	13%	35%
	32	70%	53%	7%	49%	6%	36%	57%	18%	12%	27%	13%	35%
	64	70%	53%	7%	49%	6%	36%	57%	18%	12%	27%	13%	35%
200	8	73%	94%	3%	77%	2%	58%	55%	34%	4%	43%	4%	42%
	16	70%	94%	3%	77%	2%	58%	55%	36%	4%	43%	4%	41%
	32	70%	94%	3%	77%	2%	58%	55%	36%	4%	43%	4%	41%
	64	70%	94%	3%	77%	2%	58%	55%	36%	4%	43%	4%	41%
300	8	66%	100%	1%	75%	1%	67%	51%	45%	2%	55%	2%	67%
	16	66%	100%	1%	75%	1%	67%	51%	47%	2%	55%	2%	67%
	32	66%	100%	1%	75%	1%	67%	51%	47%	2%	55%	2%	67%
	64	66%	100%	1%	75%	1%	67%	51%	47%	2%	55%	2%	67%

Results show also the effects of the location of the attacker into the MPSoC. Location (5) obtained the worst values for efficiency but the best values for efficacy when compared to the attacker in (6). Since the attacker in (5) was closer to the source of sensitive packets, more sensitive packets can be observed. On the other hand, as this attacker was in the middle of the system, the contention on the attacker communication was noisy. As a consequence a higher amount of false-positives are produced, thus reducing the attack efficiency.

Regarding the impact of the buffer size on the attack, the increase of size turns difficult to the attacker the evaluation of its injection latency. However, as the latency of sensitive packets is higher than the time to consume the packets, the impact of them in the attack was minimal.

7 CONCLUSION

NoC timing attacks are practical. In this paper, we presented an exploration of the impact of the MPSoC organization on the NoC timing attack. Four parameters were explored: i) memory organization; ii) variation of the network interface buffer sizes; iii) location of the attacker; and iv) attack threshold. We present two metrics (efficacy and efficiency) to evaluate the attacks. Experimental results demonstrate that memory organization has a huge impact on the security of the MPSoC. The configuration of the MPSoC should be used to provide performance and security properties.

ACKNOWLEDGMENTS

This work was funded by the German Federal Ministry of Education and Research (BMBF), grant number 01IS160253 (ARAMiS II).

REFERENCES

[1] 2015. Project Bonfire Network-on-Chip. https://github.com/Project-Bonfire. (2015).

[2] Daniel J. Bernstein. 2005. Cache Timing Attacks on AES. Available at: https://cr.yp.to/antiforgery/cachetiming-20050414.pdf. (April 2005).

[3] Andrey Bogdanov, Thomas Eisenbarth, Christof Paar, and Malte Wienecke. 2010. Differential Cache-Collision Timing Attacks on AES with Applications to Embedded CPUs. In *Topics in Cryptology - CT-RSA 2010*, Josef Pieprzyk (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 235–251.

[4] Eric Chien and Péter Ször. 2002. Blended Attacks Exploits, Vulnerabilities and Buffer-Overflow Techniques in Computer Viruses. In *In Proc. of Virus Bulletin Conf.*

[5] Joan Daemen and Vincent Rijmen. 2002. *The Design of Rijndael*. Springer-Verlag, Berlin, Heidelberg.

[6] J. Flich and J. Duato. 2008. Logic-Based Distributed Routing for NoCs. *IEEE Computer Architecture Letters* 7, 1 (2008), 13–16.

[7] Zecheng He and Ruby B. Lee. 2017. How Secure is Your Cache Against Side-channel Attacks?. In *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-50 ́17)*. ACM, New York, NY, USA, 341–353. https://doi.org/10.1145/3123939.3124546

[8] Paul Kocher, Jann Horn, Anders Fogh, , Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. 2019. Spectre Attacks: Exploiting Speculative Execution. In *40th IEEE Symposium on Security and Privacy (S&P'19)*.

[9] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Anders Fogh, Jann Horn, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. 2018. Meltdown: Reading Kernel Memory from User Space. In *27th USENIX Security Symposium (USENIX Security 18)*.

[10] C. Reinbrecht, B. Forlin, A. Zankl, and J. SepÃžlveda. 2018. Earthquake – A NoC-based optimized differential cache-collision attack for MPSoCs. In *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*. 648–653. https://doi.org/10.23919/DATE.2018.8342090

[11] C. Reinbrecht, A. Susin, L. Bossuet, and J. SepÃžlveda. 2016. Gossip NoC – Avoiding Timing Side-Channel Attacks through Traffic Management. In *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. 601–606. https://doi.org/10.1109/ISVLSI.2016.25

[12] C. Reinbrecht, A. Susin, L. Bossuet, G. Sigl, and J. SepÃžlveda. 2016. Side channel attack on NoC-based MPSoCs are practical: NoC Prime+Probe attack. In *2016 29th Symposium on Integrated Circuits and Systems Design (SBCCI)*. 1–6. https://doi.org/10.1109/SBCCI.2016.7724051

[13] J. Sepúlveda, D. FlÃşrez, M. Soeken, J. P. Diguet, and G. Gogniat. 2016. Dynamic NoC buffer allocation for MPSoC timing side channel attack protection. In *2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS)*. 91–94. https://doi.org/10.1109/LASCAS.2016.7451017

[14] J. Sepúlveda, M. Gross, A. Zankl, and G. Sigl. 2017. Exploiting Bus Communication to Improve Cache Attacks on Systems-on-Chips. In *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. 284–289. https://doi.org/10.1109/ISVLSI.2017.57

[15] M. J. Sepúlveda, J. P. Diguet, M. Strum, and G. Gogniat. 2015. NoC-Based Protection for SoC Time-Driven Attacks. *IEEE Embedded Systems Letters* 7, 1 (March 2015), 7–10. https://doi.org/10.1109/LES.2014.2384744

[16] Suleyman Tosun, Ozcan Ozturk, Erencan Ozkan, and Meltem Ozen. 2015. Application mapping algorithms for mesh-based network-on-chip architectures. *The Journal of Supercomputing* 71, 3 (01 Mar 2015), 995–1017. https://doi.org/10.1007/s11227-014-1348-x

[17] Yao Wang and G. Edward Suh. 2012. Efficient Timing Channel Protection for On-Chip Networks. In *Proceedings of the 2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip (NOCS '12)*. IEEE Computer Society, Washington, DC, USA, 142–151. https://doi.org/10.1109/NOCS.2012.24