Gossip NoC – Avoiding Timing Side-Channel Attacks through Traffic Management

Cezar Reinbrecht, Altamiro Susin, Lilian Bossuet, Johanna Sepúlveda

Gossip NoC - Avoiding Timing Side-Channel Attacks through Traffic Management

Cezar Reinbrecht, Altamiro Susin
Instituto de Informática
PPGC - UFRGS
Porto Alegre, Brazil
cezar.reinbrecht@inf.ufrgs.br

Lilian Bossuet
Lab. Hubert Curien
UMR CNRS 5516 - Univ. of Lyon
Saint-Etienne, France
lilian.bossuet@univ-st-etienne.fr

Johanna Sepúlveda
Inst. for Security in Information Technology
Technical University of Munich
Munich, Germany
johanna.sepulveda@tum.de

***Abstract*—The wide use of Multi-processing systems-on-chip (MPSoCs) in embedded systems and the trend to increase the integration between devices have turned these systems vulnerable to attacks. Malicious software executed on compromised IP may become a serious security problem. By snooping the traffic exchanged through the Network-on-chip (NoC), it is possible to infer sensitive information such as secrets keys. NoCs are vulnerable to side channel attacks that exploit traffic interference as timing channels. When multiple IP cores are infected, they can work coordinately to implement a distributed timing attack (DTA). In this work we present for the first time the execution of a DTA and a secure enhanced NoC architecture able to avoid the timing attacks. Results show that our NoC proposal can avoid the DTA with an increase of only 1% in area and 0.8% in power regarding the whole chip design.**

***Index Terms*—Network-on-Chip, Security NoC, Timing Attack, Timing Side-channel Attack.**

I. Introduction

Nowadays, MPSoCs are object of attacks. One of the most dangerous attacks is the side-channel attack (SCA) [1] [2]. It can be performed remotely by inferring sensitive information through architecture behavior leakage, such as timing, scheduling or communication addressing. Among these attacks, the timing attack is most common [3] [4] [5].

When a sensitive application is mapped into an MPSoC, it creates several sensitive communications between the IPs involved. For example, one processing element that has sensitive data requires to a secure processor to encrypt or decrypt such information. Then, during the cipher algorithm execution, this IP has to make several accesses to the shared memory, in order to retrieve the pre-computed values used in such algorithms. And, if there is a miss, the shared memory has to request data for the main memory. In NoC-based MPSoCs, it is not possible to directly access the information transmitted between the IPs, not being able to perform attacks that listen the channel. Therefore, an attacker needs to use indirect information that reveals the system behavior. The timing attack in NoC aims to infect an IP to sample the injection delay of its own packets in a path that intersects a sensitive communication. Then, the degradation of its own throughput can reveal the presence of sensitive traffic. Therefore, knowing the access times of the sensitive communication and the cipher algorithm used, the attacker can infer the look-up tables accessed in memories in order to discover the secret key [6].

When multiple IP cores are infected, they can work coordinately to implement a distributed timing attack (DTA). DTA threat model requires that the attacker controls data injection of the malicious IP and knows some characteristics of the MPSoC system. Hence, the attacker can force its communication flow to share the same NoC resources than the sensitive flow. Although this kind of attack is object of recent research, DTA has not been demonstrated in MPSoCs yet. However, the threat is real and new security mechanism must be developed [7]. Network-on-Chip (NoC), as the communication structure of the MPSoC, plays an important role regarding systems security. Its main function in the system operation allows the NoC to detect and avoid attacks. Previous works addressed the timing attack protection by security enhanced NoCs [7] [8] [9]. They show that NoC can detect and protect the systems against this kind of attacks. However, such approaches decrease the overall system performance and they were not tested under a DTA.

In this work we propose *Gossip NoC*, a security enhanced NoC able to identify traffic anomalies and handle through a distributed management scheme. Each router has a traffic monitor to identify when there is a possible threat and send a *gossip* alert. The neighbor routers receive the *gossip* message, where each router can reinforce or ignore the message. Reinforcement strategy reduces false positive detections. If an attack is detected, the router changes the routing algorithm for the packets that uses that path. Our work presents three main contributions:

1) A DTA execution in an MPSoC environment;
2) a secure enhanced NoC architecture able to protect against DTA; and
3) the *gossip* reinforcement strategy able to avoid false positives attacks.

This paper is divided in six sections. Section II presents the related works, regarding NoC protection mechanisms. Timing Side Channel Attacks are presented in Section III. The Gossip NoC proposal is presented in Section IV and the experiments and results are shown in Section V. Finally, we conclude the paper in Section VI.

II. Related Works

Previous works [10] [11] [12] show that the NoC can be an effective protection mechanism to the system. By monitoring and controlling the data exchange inside the MPSoC, the NoC can detect and avoid software-based attacks. Previous works [7] [8] [9] and [13] addressed the NoC-based protection against software-based SCA. Such works propose to modify the arbitration and routing of sensitive traffic.

The works of [7] [8] propose the integration of hard QoS mechanism to isolate the sensitive information. They include temporal network partitioning, based on high [7] and bounded [8] priorities arbitration schemes. Furthermore, the work of [9]

proposes random arbitration and adaptive routing as protection techniques. On the other hand, multiple path communication for sensitive flows is proposed by [13]. Despite their good results, these previous works have two main drawbacks: i) QoS approaches decrease the overall system performance; ii) the system keep vulnerable to Denial-of-Service (DoS) attacks; they are only capable to protect a system where only a single attacker, located in the middle of the sensitive path, is saturating the NoC traffic and sniffing the sensitive flow.

In order to overcome such drawbacks, in this work we propose a distributed security mechanism, able to deal with timing attack while introducing a lower impact on system performance. Our approach handles distributed attacks, such as performed by DTA, and also introduces a strategy to avoid the false-positives detections.

III. Distributed Timing Attack (DTA)

A timing attack is a side-channel analysis that exploits information leakage through computation or communication timing behavior. Previous works have already shown methods to attack computational and storage components by exploring features such as cache miss rates [14] [15] or the time required to perform floating point operations [16].

The work of [9] presented a scenario where a malicious IP inside the MPSoC injects data to saturate a NoC path. As a result, the attacker is able to measure the degradation of its own throughput, in order to infer information about the traffic that intersects the attackers router. Hence, the attacker can discover the access pattern of a sensitive information. The work of [9] demonstrated the SCA through a system that tracks the communication between a crypto-processor and its cache memory. In order to improve the effectiveness of the attack, multiple cores can be infected. Compared to the single timing attack, the DTA decreases the computation and storage requirements of the malicious IPs. This work presents, for the first time, a DTA at NoC, where it divides the malicious IPs into two groups:

- *Injectors*, responsible for injecting data in the NoC at high data rates with the objective to increase the congestion of the attacked path;
- *Observers*, responsible to inject data at lower data rates and to sample the delay of their own packets with the objective to collect the throughput traces of the attacked path.

Besides, the malicious packets do not necessarily violate the security policies of the NoC (source/destiny/size) to execute the attack, so often these packets can not be detected by a secure enhanced NoC. The following subsections will present the threat model and the three stages to develop such attack: i) infection, which inserts a malicious software (malware) in the target system; ii) calibration, used to adjust the attack parameters; and iii) execution and cryptanalysis, which develops the attack, data collection and post-processing.

A. Threat Model

In order to execute a DTA in a NoC, the following conditions are required:

1) The attacker can infect more than two processors of the MPSoC through a malicious software.
2) The attacker knows where the target elements are located in the NoC (logical address).

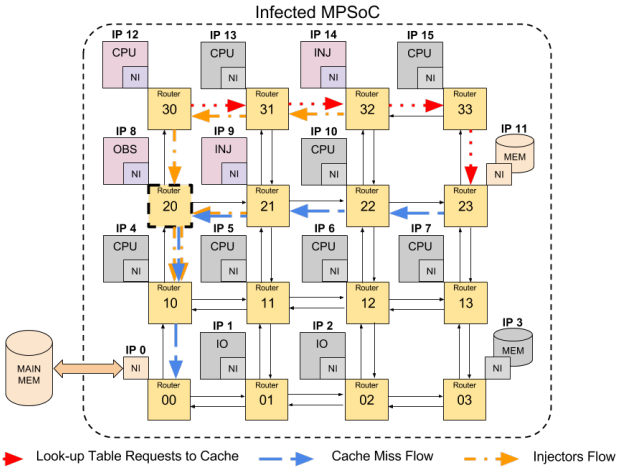

Fig. 1: MPSoC system running a sensitive application, after infection stage.

3) The attacker knows the topology and routing algorithm of the communication infrastructure.
4) The attacker observation path intersects the sensitive path.
5) Infected IPs are close (few hops) to the sensitive path.

B. First Stage - Infection

The purpose of this step is to spread a malicious application into MPSoC. This application uses the infected IP to transmit the malware and to create the two types of malicious IP - *injector* and *observer*. Figure 1 presents an MPSoC system composed of CPUs, shared memories, peripherals (IO) and one main memory access; all integrated through a 2D mesh NoC. It also shows three communication flows, two regarding a sensitive application and one regarding the attacker flow.

The sensitive application starts with an encryption/decryption running on the CPU (IP 12). This CPU accesses a shared memory (IP 11), in order to retrieve the look-up tables used in the cipher algorithm. When there is a cache miss at this memory, it creates another communication flow requesting data for the main memory, addressed as IP 0. Also, Figure 1 presents three malicious IPs, one *observer* at IP 8 (labeled as *OBS*) and two *injectors* at IP 9 and 14 (labeled *INJ*). These malicious IPs will create the attacker communication flow to IP 4, a normal CPU, with the objective to congest the path used for cache misses requests.

C. Second Stage - Calibration

The work in [17] presents an exploration of the communication behavior in a NoC-based MPSoC over several applications from SPLASH-2 and PARMACS benchmarks. According [17], the synchronization-intensive applications can reach 70% of injection rate. Thus, NoCs designers configure the communication structure to meet the requirements, and avoid throughput penalties. In this case, DTA have to congest the target path at least at 70% of the bandwidth. Then, the normal traffic will use the last 30% of the available bandwidth and influence directly the throughput of the *observer*.

Another important issue is that the sensitive information is characterized by long packets [18]. This means that during

a burst communication of a sensitive packet in the NoC, a significant part of the bandwidth is consumed. This value depends on the cipher block sizes and NoC enlace size, and at our environment it can represent 10% of the bandwidth. This means that the sensitive packets degrade the throughput more abruptly than a normal packet. Therefore, the objective of the calibration is to define a DTA configuration that creates a congestion high enough to put in evidence any packet in the target path, but lower enough to distinguish between a normal and sensitive packet.

The calibration stage is performed exploring different attack scenarios, varying the number of *injectors* and the injection data rate. In each scenario, the *injectors* have to respect an injection time t_i and a waiting time t_w, used to guarantee a clear observation of the contribution of the *injectors* in the *observer* throughput. It is performed until the mean throughput during t_i has a low variance. Figure 2 presents a part of four calibration scenarios merged in the same time line. The mean throughput of each case is represented by the black dash. On the left side of the Figure 2, three throughput zones are presented:

- Saturation: Difficulty to distinguish the normal and sensitive packets. Above 90% of the bandwidth.
- Congestion: Different behaviors of throughput can be observed (big and small packets). Between 70% and 90% of the bandwidth.
- Traffic not intense: Difficulty to identify the falls of throughput, because collision of the packets hardly occurs. Bellow 70% of the bandwidth.

Four calibration were performed and can be observed in the Figure 2. The first scenario, Figure 2(a), used three *injectors* injecting packets at 50% of data rate and one *observer*. The second, Figure 2(b), used three *injectors* at 30% of data rate. Since the throughput presented in Figure 2(a) and (b) was in the saturation zone, the next two calibrations reduced the number of *injectors*. The third scenario, Figure 2(c), used two *injectors* at 50% of data rate; and Figure 2(d), used two *injectors* at 30%.

The scenario of Figure 2(d) is the best configuration for the attack, because it degrades the throughput at lower levels but stays in the congestion zone. This feature allows the observer to detect when a sensitive packet is passing through the path. The throughput choosen in the calibration step is used as the threshold of the attack. Any throughput result at execution stage bellow the threshold will be considered as a possible sensitive packet.

The t_i and t_w times of the calibration step depend on the NoC characteristics. High capacity NoCs require more time to achieve saturation, that is to fill the buffers and to generate a congestion.

D. Third Stage - Execution and Cryptanalysis

Once calibrated, the *injectors* will send packets, the *observer* will sample and store the throughput results. Regarding the execution of DTA, there are several types of attacks that can be made through the timing analysis. In this work, we target the strategy to infer the access pattern of a memory that contains the lockup tables of the cryptographic IP. This is the same attack strategy presented by [9], but we extend the approach to a distributed attack. On this scenario, DTA requires the *injectors* configured with the values determined at

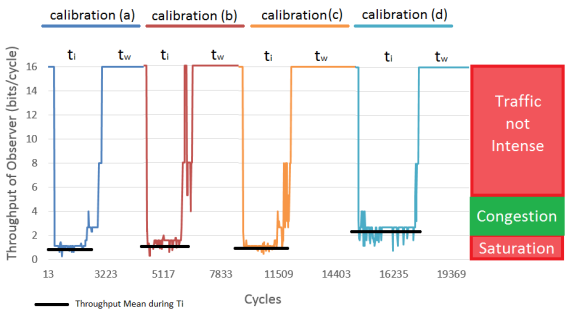

Fig. 2: Throughput Trace from observer attacker. Four calibration scenarios: a) Three injectors at 50%; b) Three injectors at 30%; c) Two injectors at 50%; d) Two injectors at 30%.

calibration step, and only one *observer* to collect the memory throughput results.

The last part of DTA execution regards to the cryptanalysis, which employs a mathematical algorithm to correlate the timing results collected by the monitor to infer an unknown key. Some techniques are demonstrated at [6], [19] and [20].

IV. Gossip Network-on-Chip

In this work we propose the *Gossip NoC*, a security enhanced architecture to deal with single and distributed timing attacks. It is composed by a traffic monitor and a countermeasure technique at each router, being a distributed security mechanism. The name *gossip* is proposed because the traffic monitors generate alert messages to other routers, creating a *gossip message*. Besides, in order to avoid false-positives the router uses a reinforcement parameter, called *gossip confidence*, to decide when to accept the *gossip message*. If an attack is detected, the router changes the routing algorithm to the packets that wants to go through the path under attack. Therefore, *Gossip NoC* combines two strategies to protect the MPSoC against DTA: i) Detection, which includes the bandwidth monitoring and the *gossip message* generation in the presence of an abnormal behavior; and ii) protection, triggered when any *gossip message* is received and which is able to modify the route of the packet.

A. Gossip Architecture

The *gossip router* microarchitecture is shown in Figure 3. It is based on a typical NoC router composed by routing scheme (XY and YX), Round-Robin arbiter and FIFO memory. Additional three main components are integrated:

- Gossip In Block (1) : It controls the internal state of the *gossip router* according the values of the input signals (outputGossip). When the number of *gossip messages* received from neighbor routers overcomes the *gossip confidence*, an attack is confirmed. As a result the routing of the packets is modified.
- Gossip Logic (2) : It commutes the incoming packets. Under attack, the traffic is commuted according to the YX algorithm (RouteType). Otherwise the XY route is implemented.
- Gossip Generator (3) : It monitors the traffic bandwidth. When it exceeds a protection bandwidth threshold, a

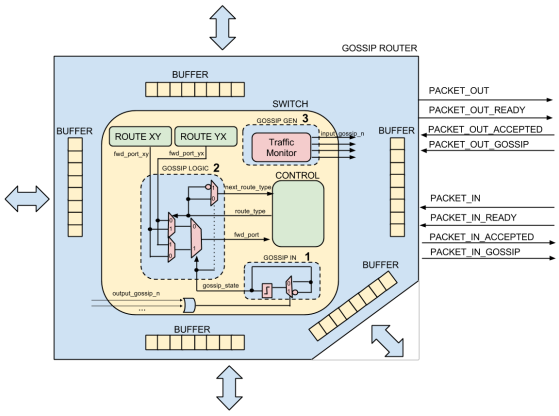

Fig. 3: *Gossip router* microarchitecture: (1) Gossip In Block; (2) Gossip Logic; (3) Gossip Generator.

signal indicating a possible attack is activated and transmitted (inputGossip) to the *Gossip In Block* of all the neighbor routers. This configures a *gossip message*.

B. Gossip Functionality

Gossip router modifies the XY routing algorithm to the YX in the presence of an attack. As a result, the malicious IP that was collecting information (*observer*) in such a path is now unable to complete the attack, since the sensitive information is using another route. The usage of XY and YX routing algorithms together is guaranteed as deadlock and livelock free [21] [22].

However, attackers may notice the change of routing algorithm. In order to overcome such a drawback, it is required that the MPSoC limits the number of processing elements that can execute external tasks, and thus the possible attackers. As *Gossip NoC* changes the route during runtime, the attacker needs to create, simultaneously, two zones of congestion (one for each possible route). Thus, requiring several infected IPs to accomplish this goal.

During design phase, two parameters have to be adjusted, the throughput threshold and the *gossip confidence*. The throughput value triggers a *gossip message* to the neighbor routers. The *gossip confidence* defines the minimum number of *gossip messages* required to activate the countermeasure mechanism (routing algorithm modification).

V. Experimental Results and Analysis

The target MPSoC is composed by eight processing elements and eight shared memories, as shown in Figure 1, interconnected by a 4x4 mesh-based NoC. The NoC is composed by a 5-port router, with input buffers of eight flits, each flit of 32 bits, and using an XY as routing algorithm.

Our experimental environment implements the DTA to measure the access times from a crypto-processing element and its cache memory. As presented in [9], the objective of this attack is to understand the access pattern of the memory that stores the look-up tables with the pre-computed data used in a cryptographic algorithm. This first experiment demonstrates the effectiveness of DTA attack with an unprotected NoC. Then, the second experiment presents protection against DTA when *Gossip NoC* is used. The system was modeled and simulated by the VHDL framework presented in [23]. The unprotected NoC and *Gossip NoC* were synthesized to a 65nm ASIC technology, in order to obtain information of overhead in power and area.

A. DTA Experiment

The DTA performed the three stages described in Section 3. The calibration, depicted in Figure 2, presents four possible scenarios, where (d) was chosen as the suitable value for this attack. It has a mean throughput of 2.23 bps (bits per cycle) during the injection times (t_i). This value is used during attack execution as the threshold throughput to detect possible sensitive packets. Any throughput sample below 2.23 bps is interpreted by the attacker as a possible memory access of the crypto-processor.

The simulation used two *injectors* at a rate of 30%, the *observer* at 15%, the crypto-processor at 10% and other IPs in the system with random targets at 10% (communication noise). Simulation results are presented in Figure 4 with 50,000 traces. It can be observed that in the first 3,000 samples the throughput is high, at 16 bps, and after it falls to around 3 bps. This phenomenon is called in this work as "warm-up", which means that in the beginning of the simulation all buffers are empty, and until they fill, the throughput does not fall. Also, in Figure 4 a dashed line is presented, representing the threshold of 2.23 bps.

Table I presents the effectiveness, success rate, false-positive (FP) and false-negative (FN) results of the *observer* attacker measures during DTA. The effectiveness is the quantity of real sensitive data identified by the attack correctly, in this experiment about 59%. The success rate of the attack is the percentage of the points below the threshold that corresponds to a sensitive data. A success rate of 97% means that almost all points below the threshold correspond to the sensitive data, being a very reliable information. This reflects the results of FP, where only 3% of the identified points were a mistake. The FN represent the part of the sensitive information invisible to the attacker. These results used an error margin of one packet transmission to deal with a variance between the sample and sensitive packet, and the warm-up phase was not considered.

The protection strategies against timing attacks, such as random arbitration and adaptive routing proposed by [9]

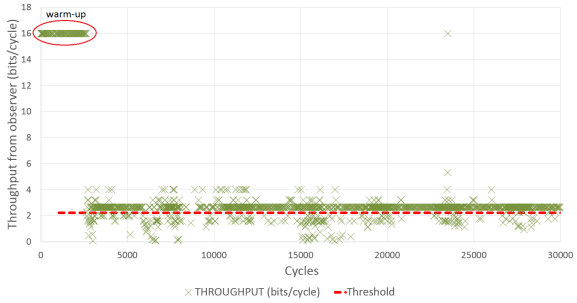

Fig. 4: Trace throughput of the DTA on the memory access pattern. Threshold identified in the dashed line.

TABLE I: Effectiveness (% of matches) of DTA using a threshold of 2.23 bps.

	Effectiveness	Success Rate	FP	FN
DTA	59.63%	96.92%	3.07%	40.36%

are not effective against DTA. Random arbitration protection highly depends on the randomness degree. Observers can disclosure the pseudo-random sequence of the arbiter during the cryptanalysis stage. Adaptive routing protection can be also avoided if multiple observers are operating simultaneously in the possible paths. Therefore, the distributed aware protection mechanisms becomes mandatory.

B. Gossip NoC Experiment

The same DTA attack and configuration was developed in the system with the *Gossip NoC*. The simulation results are presented in Figure 5. The warm-up phase can be observed in the firsts 3000 cycles. After the warm-up, the throughput falls to 2.5 bps, being a similar behavior from DTA attack without protection. However, after cycle 4000 the behavior of the throughput changes again, bringing the mean throughput lower than the threshold, marked in Figure 5 as the first anomaly detection. Then, after the cycle 15000 a new behavior occurs, where the mean throughput increases above the threshold, marked as the second anomaly detection. This indicates that *Gossip NoC* identified anomalies (high bandwidth usage) in the traffic and changed two times the route of neighbor packets (at cycle 4000 and at cycle 15000), altering all communications behavior.

The first changing in behavior (at cycle 4000) created an increase of communication above expected from *observers* router. This happened because the *gossip routers* detected not only the attackers traffic as anomalies, but the normal ones too. So, routers near the *observer* were triggered with *gossip messages* and changed the route. Then, because of this increase, at time 15000, the *observer* route identified this intense traffic as another threat, creating new *gossip messages*, and reducing the congestion (increasing the throughput) as a consequence.

In this experiment, what really happened with the sensitive packets was that they were rerouted at cycle 6000. This information can be confirmed with the results presented in Table II. It confirms the fact that these packets changed their routes, not passing in observer router, because 91% of the sensitive data was not identified, achieving an effectiveness of only 6.62%. The success rate of the attack decreased to 7% due to the traffic noise inserted by the *gossip router* mistakes, that change the route of normal packets around, inserting wrong data in observer measurements. So, except these 7% of attack success rate, the throughput degradation was not representing sensitive information.

The reason that normal packets were considered a threat by the *Gossip NoC* is because the *gossip confidence* parameter used was too low. It used the value 1 as parameter, where this value refers the quantity of *gossip messages* required to consider a threat. This means that even high used paths by normal packets could be considered anomalies by the system. To avoid these false alarms, the *gossip confidence* has to be adjusted, in this case, increased. Therefore, another experiment with Gossip NoC was developed, increasing this parameter, in order to understand this reinforcement strategy potentials.

The result of the experiment with a *gossip confidence* of ten (10 *gossip messages* to believe that is a threat) are presented in the Figure 6. The throughput trace depicted seems to be the same from a typical DTA attack, but the results of effectiveness in Table III demonstrates that the protection mechanism was effective. The throughput trace did not show changing in behavior because the Gossip identified only the attacked path, and rerouted successfully the sensitive path. However, increase the *gossip confidence* from one to ten allows the attacker to double its effectiveness, from 6% to 13%. This happened because the *Gossip NoC* needed more time to reinforce the fact that there was an attack, given more time to the attacker. Therefore, *gossip confidence* presents a trade-off between the impact on the normal behavior of the system and the protection effectiveness.

In order to understand this trade-off, Figure 7 presents the results of DTA effectiveness when changing the *gossip confidence* parameter from 1 to 100. Simulations executed changing this parameter with a step of 1 from 1 until 20, and after that the step was 10. The last point represents the value where there is no protection, because the DTA achieves its normal effectiveness of 59%. It can be observed that the increase of *gossip confidence* decreases the protection linearly. However, as presented in the Figure 6, a *gossip confidence* of 10 already has no much impact in communications behavior. Therefore, it should never be a value near 1, because it creates much false-positive detections, but neither a high one, because it decreases the protection level. The proper value has to be a lower value, from 10 to 40, in order to maintain the attack

TABLE II: Effectiveness (% of matches) of DTA using a threshold of 2.23 bps under *Gossip NoC*.

	Effectiveness	Success Rate	FP	FN
DTA	6.62%	6.56%	93.43%	93.37%

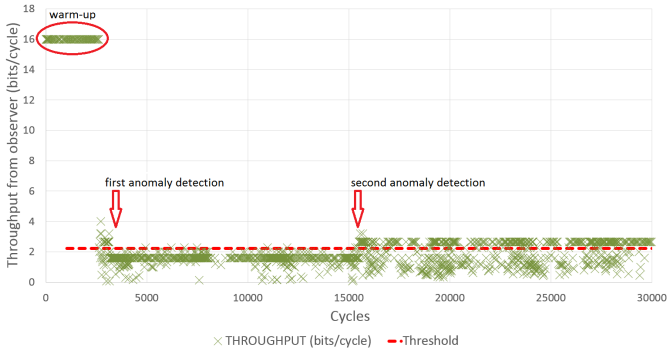

Fig. 5: Trace throughput of the distributed timing attack under *Gossip NoC*. *Gossip confidence* of 1.

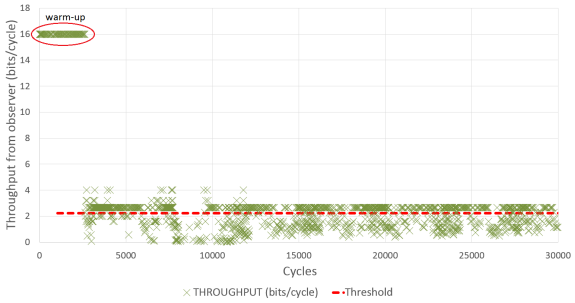

Fig. 6: Trace throughput of the DTA under *Gossip NoC*. *Gossip confidence* of 10.

TABLE III: Effectiveness (% of matches) of DTA using a threshold of 2.23 bps under *Gossip NoC*. *Gossip confidence* of 10.

	Effectiveness	**Success Rate**	**FP**	**FN**
DTA	13.55%	19.06%	80.93%	86.44%

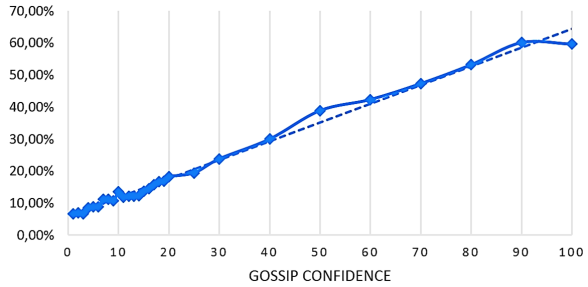

Fig. 7: Effectiveness of DTA for 50000 traces according different *gossip confidence* configurations.

TABLE IV: Synthesis Results for the Typical and Gossip router.

	Unprotected NoC	Gossip NoC	% Overhead
Area (um2)	2632	3189	21.16%
Power (mW)	2.073	2.409	16.2%

effectiveness below 30%, which means, to avoid that the attack acquires much data.

C. Synthesis Results

Table IV presents synthesis results of a unprotected router and *Gossip router* under 65 nm technology and 1GHz as reference clock frequency. Results show that each gossip router has 21% area and 16% power overhead when compared to the typical router. In spite of the high percentage, the interconnection network represents only 5% of the whole MPSoC system, being the Gossip NoC an increase of 1% in area and 0.8% in power regarding the whole chip design.

VI. Conclusion

This paper presents the *Gossip NoC*, a security enhanced NoC architecture able to protect MPSoCs against DTA. By using distributed monitor and traffic management, it is possible to detect possible attacks and deflect the sensitive traffic. The proposed architecture was capable to avoid DTA, since the sensitive data was reflected, resulting in a reduction of effectiveness from 59.63% to 6.62% in a 50000 traces scenario. Besides, for the first time it was presented a NoC-based mechanism able to avoid false positives attacks by using *gossip confidence* threshold. Results showed that a difference of 10 times in this parameter can reduce drastically the false alarms in the system, but also allows the attacker to acquire double of information. *Gossip NoC* presented a low cost architecture solution with an increase of 1% in area and 0.8% in power regarding the whole chip design.

References

[1] R. Karri, K. Wu, P. Mishra, and Y. Kim, "Concurrent error detection of fault-based side-channel cryptanalysis of 128-bit symmetric block ciphers," in *Design Automation Conference, 2001. Proceedings*, 2001, pp. 579–584.

[2] P. Bayon, L. Bossuet, A. Aubert, V. Fischer, F. Poucheret, B. Robisson, and P. Maurine, "Contactless electromagnetic active attack on ring oscillator based true random number generator," May 2012, pp. 151–166.

[3] P. Zhou, T. Wang, G. Li, F. Zhang, and X. Zhao, "Analysis on the parameter selection method for flush+reload based cache timing attack on rsa," *Communications, China*, vol. 12, no. 6, pp. 33–45, June 2015.

[4] J. Wang, Y. Yang, L. Chen, G. Yang, Z. Chen, and L. Wen, "A combination of timing attack and statistical method to reduce computational complexities of ssl/tls side-channel attacks," in *Computational Intelligence and Security (CIS), 2015 11th International Conference on*, Dec 2015, pp. 402–406.

[5] U. Herath, J. Alawatugoda, and R. Ragel, "Software implementation level countermeasures against the cache timing attack on advanced encryption standard," in *Industrial and Information Systems (ICIIS), 2013 8th IEEE International Conference on*, Dec 2013, pp. 75–80.

[6] D. Jayasinghe, J. Fernando, R. Herath, and R. Ragel, "Remote cache timing attack on advanced encryption standard and countermeasures," in *Information and Automation for Sustainability (ICIAFs), 2010 5th International Conference on*, Dec 2010, pp. 177–182.

[7] W. Yao and E. Suh, "Efficient timing channel protection for on-chip networks," in *Networks on Chip (NoCS), 2012 Sixth IEEE/ACM International Symposium on*, 2012, pp. 142–151.

[8] H. Wassel, G. Ying, J. Oberg, T. Huffmire, R. Kastner, F. Chong, and T. Sherwood, "Networks on chip with provable security properties," *Micro, IEEE*, vol. 34, no. 3, pp. 57–68, May 2014.

[9] M. Sepulveda, J.-P. Diguet, M. Strum, and G. Gogniat, "Noc-based protection for soc time-driven attacks," *Embedded Systems Letters, IEEE*, vol. 7, no. 1, pp. 7–10, March 2015.

[10] J.-P. Diguet, S. Evain, R. Vaslin, G. Gogniat, and E. Juin, "Noc-centric security of reconfigurable soc," in *Networks-on-Chip, 2007. NOCS 2007. First International Symposium on*, May 2007, pp. 223–232.

[11] L. Fiorin, G. Palermo, and C. Silvano, "A security monitoring service for nocs," in *Proceedings of the 6th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, ser. CODES+ISSS '08, 2008, pp. 197–202.

[12] L. Fiorin, S. Lukovic, and G. Palermo, "Implementation of a reconfigurable data protection module for noc-based mpsocs," in *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, April 2008, pp. 1–8.

[13] R. Stefan and K. Goossens, "Enhancing the security of time-division-multiplexing networks-on-chip through the use of multipath routing."

[14] F. Liu and et all, "Last-level cache side-channel attacks are practical," in *Security and Privacy (SP), 2015 IEEE Symposium on*, 2015, pp. 605–622.

[15] S. Crane, A. Homescu, S. Brunthaler, P. Larsen, and M. Franz, "Thwarting cache side-channel attacks through dynamic software diversity," in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2014*, 2015.

[16] M. e. a. Andrysco, "On subnormal floating point and abnormal timing," in *Security and Privacy (SP), 2015 IEEE Symposium on*, 2015, pp. 623–639.

[17] M. Monchiero, G. Palermo, C. Silvano, and O. Villa, "Exploration of distributed shared memory architectures for noc-based multiprocessors," in *IC-SAMOS 2006.*, July 2006, pp. 144–151.

[18] C. Mancillas, M. Mendez Réal, L. Bossuet, G. Gogniat, V. Fischer, and A. Baganne, "Extending Multicore Architectures with Cryptoptocessors and Parallel Cryptography," in *Colloque national du GDR SOC-SIP*, Paris, France, Jun. 2014. [Online]. Available: https://hal-ujm.archives-ouvertes.fr/ujm-01015264

[19] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," in *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, ser. CRYPTO '96, 1996, pp. 104–113.

[20] Y. Tsunoo, T. Saito, T. Suzaki, M. Shigeri, and H. Miyauchi, *Cryptographic Hardware and Embedded Systems - CHES 2003: 5th International Workshop, Cologne, Germany, September 8–10, 2003. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, ch. Cryptanalysis of DES Implemented on Computers with Cache, pp. 62–76.

[21] A. Borhani, A. Movaghar, and R. Cole, "A new deterministic fault tolerant wormhole routing strategy for k-ary 2-cubes," in *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*, Dec 2010, pp. 1–7.

[22] K. Tatas, S. Sawa, and C. Kyriacou, "Low-cost fault-tolerant routing for regular topology nocs," in *Electronics, Circuits and Systems (ICECS), 2014 21st IEEE International Conference on*, Dec 2014, pp. 566–569.

[23] J. Sepulveda, R. Pires, G. Gogniat, W. J. Chau, and M. Strum, "Qoss hierarchical noc-based architecture for mpsoc dynamic protection," *Int. J. Reconfig. Comput.*, vol. 2012, pp. 3:3–3:3, Jan. 2012.