SecureNoC: Enhancing On-Chip Network Security for Many Integrated Core Systems

Xiaoming Guo, Sijia He, Amlan Nayak, and Jay Zhang
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI
EECS 578 Fall 2015 Final Report

***Abstract*—With the unimpeded scaling of transistor technology, multiprocessors with a large number of integrated cores on a single die are emerging as efficient and practical solutions for the warehouse-scale computing and datacenter markets. Enabling scalable data communication between such cores is a challenge. Networks on Chip (NoCs) have emerged as the most promising and readily implementable solution. Architected with a blueprint that shares many similar design aspects with existing macro-scale networks such as the Internet, NoCs face similar security vulnerabilities. In this paper, we address two such vulnerabilities, namely Denial of Service (DoS) attacks and unauthorized secret-information extraction. In order to detect and recover from DoS attacks, we implement an anomalous high-packet injection rate detector and node-blocking mechanism to detect and recover from single or multi-node based DoS attacks. Secondly, we introduce the Secure Packet ExchAngeR (SPEAR) system which enables a foolproof method for a pair of nodes in an NoC to send and receive privileged data-packets by eliminating the possibility of intermediate nodes from eavesdropping on the communication. We demonstrate that these enhancements can be implemented as simple extensions of existing NoC architectures without any significant drawbacks in performance or area.**

I. Introduction

With the advent of chip multiprocessors with as many as 72 integrated physical cores [1], the architecture of efficient on-chip communication networks is becoming a primary design concern both in academia and in industry [2]. Increasing core counts have engendered an evolution of these networks from simple indirect links and uni-directional buses to complex point-to-point packet-switched networks with a variety of topologies and flow-control mechanisms [3]. Increasingly, these Networks-on-Chips (NoCs) are beginning to resemble macro-scale networks such as the Internet, with dedicated routers and layered communication protocols. While such complex infrastructures facilitate efficient communication and improve quality of service (QoS), they also introduce new vulnerabilities that can be exploited by hostile agents to carry out a variety of attacks [4].

Network-oriented attacks are particularly harmful as they are not isolated to a single node, but are prone to spreading across a multitude of nodes and possibly rendering the entire network inoperable [5]. In this paper, we address two particular categories of attacks: software-based Denial of service (DoS) attacks and extraction of secret-information attacks. DoS attacks can cripple a network by saturating it with a torrent of packets issued from either one or a number of compromised nodes. As a network becomes congested, data can no longer be exchanged between nodes, leading to a severe reduction in the QoS.

Secret-information extraction is a form of attack that seeks to read sensitive in-flight data while it is passing through a compromised node in a network. With the introduction of secured computing frameworks such as Intel's SGX [6] and ARM's TrustZone [7], applications can create protected "enclaves" in memory to store sensitive information which are protected even from malicious operating systems. Such sensitive data is encrypted before it is sent to off-chip DRAM memory. However, in a many-integrated core system, the data packets must be shuttled around the NoC in order to reach the memory interface. Intermediate nodes along this path have the potential to quietly ingest these packets and re-transmit them, unbeknownst to the source node. This form of eavesdropping can be broadly categorized as a small-scale man-in-the-middle attack. To the best of our knowledge, such attacks have not been addressed at the hardware level in NoCs. In this work, we present security infrastructures that can provide complete protection against both DoS and eavesdropping attacks.

The paper is organized as follows. Section II provides a brief overview of selected work that has already been published in the area of NoC security. Section III outlines in detail the design of the proposed mechanisms. Sections IV and V present the experimental setup and simulation results, respectively. This is followed by a short overview of the potential for future improvements of the presented work in section VI. We summarize and conclude in section VII.

II. Related Work

The work presented in this paper addresses NoC security issues that have been recognized as important vulnerabilities in the literature. Porquet, Greiner, and Schwarz propose a dedicated Memory Protection Unit to permit the co-hosting of a number of software stacks which can execute in a variety of security domains [8]. Their approach is a hardware/software codesign which can incur significant area and power penalties which is of particular concern in small, embedded systems, such as those being targeted by the authors. Further, they introduce an virtualization scheme that can lead to additional latency due to the added step of address translations. Lukovic and Christianos [9] direct their effort towards protecting NoCs against buffer-overflow attacks by embedding data protection

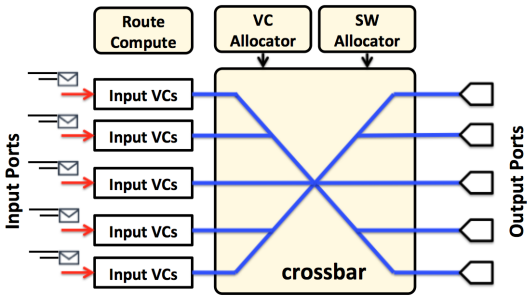

Fig. 1. An input-queued virtual channel router

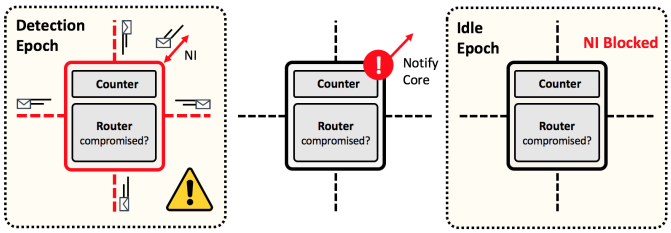

Fig. 2. DoS attack detection phase 1 (high traffic detection and temporary blocking of potentially compromised node)

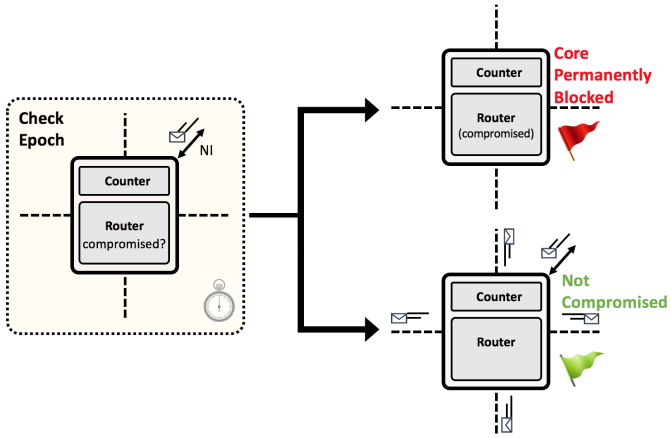

Fig. 3. DoS attack detection phase 2 (local port permanently disabled if high injection rate is detected again)

modules into the network interfaces connecting cores with routers. Their mechanism prevents the propagation of a buffer-overflow attack to other nodes in the system. Similar to [8], this solutions also results in unavoidable area and performance overheads that may not be acceptable in a commercial setting. The security mechanisms demonstrated in this paper incur negligible hardware overheads while simultaneously providing absolute protection from DoS and eavesdropping attacks. Moreover, unlike [8] no modifications need to be made to existing software stacks in order to implement the proposed infrastructure.

III. Design and Implementation

A. Network-on-Chip architecture

In this work, we use the input-queued virtual channel router as described in the textbook by Dally and Towles [10]. As shown in Figure 1, a typical router consists of input ports, output ports, route computation logic, virtual channel (VC) allocator, switch (SW) allocator, and crossbar [11].

Each port in the router consists of virtual channels to avoid deadlocks and improve network utilization. Packets are partitioned into flits to enable wormhole routing. The routing mechanism is pipelined into four distinct stages: input buffering and route computation, VC allocation, switch allocation, and crossbar traversal. Incoming flits are stored in the buffers within the virtual channels, waiting to be transferred. VC allocation assigns an output VC to the head flit of a packet and the body flits of that packet inherit the VC assigned to the head flit. The SW allocator arbitrates the requests of flit transfers from input ports to output ports. Once an input port and output port pair is granted, the crossbar will connect them together to enable data transfer.

We use a mesh network in our paper. Each router consists of five input ports and five output ports, four of which connect to neighboring routers, and the remaining one is an ingress port for the local node. Unless specified otherwise, each port has four virtual channels, and the buffer size for each virtual channel is eight.

B. Threshold-based DoS attack detection

We have implemented a threshold-based DoS attack detection mechanism. Each router within the network is augmented with a high packet injection rate detector. As shown in Figure 2, the detector counts the number of flits injected into the router from the local port during a empirically-determined number of cycles defined as an epoch. If the number exceeds the predefined threshold value, the system will temporally stop accepting more packets from the "potentially compromised" node for two epochs. In the meantime, the "potentially compromised" core is notified by the router to re-schedule the packet injection process.

Figure 3 shows the second phase of the DoS detection. After the two-epoch stalling period, the local port is re-enabled, and if the core continues to inject packets at a rate larger than the threshold, it will be blocked permanently; otherwise, the core is allowed to inject packets normally. This second chance mechanism aids in eliminating false positives.

C. SPEAR

The **S**ecure **P**acket **E**xch**A**nge**R** (**SPEAR**) supports the secure exchange of data between two nodes by ensuring that other intermediate nodes in the system cannot snoop or interfere with the transfer of privileged packets. We reserve Virtual Channel 0 (VC 0) only for privileged packets (indicated by a bit in the head flit). Non-privileged packets will never be allocated to VC 0. When privileged packets are in transit, the switch allocator will prioritize their constituent flits by only granting to VC 0, which in turn blocks all allocation requests from non-privileged packets. In addition, the ingress/egress

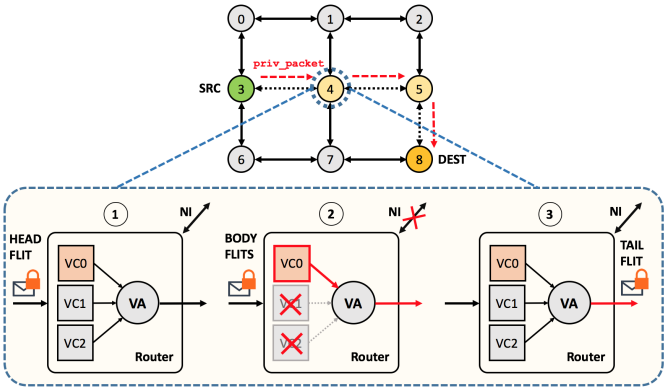

Fig. 4. SPEAR phases of operation

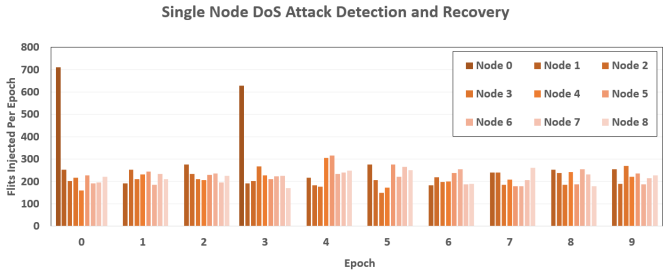

Fig. 5. Node 0 DoS attack detection and recovery

port between the router and the local core is also blocked during this period to ensure that the core cannot eavesdrop on the privileged data.

We use Figure 4 to further explain the SPEAR mechanism. In the presented situation, a privileged packet is being sent from node 3 (labeled SRC) to node 8 (labeled DST). Assuming that the network uses deterministic XY routing, we can see that the packet will traverse the path 3-4-5-8. Privileged packet propagation through intermediate router 4 is shown in the second half of the image. After the head flit of the privileged packet reaches the router, it is allocated to VC 0 at the appropriate output port. The local port is blocked to prevent any eavesdropping. During switch allocation (not drawn on figure), VC 0 has priority over all other VCs, and thus the privileged head flit will traverse the crossbar immediately. The body flits will inherit the VC allocation (VC 0) of the head flit. Again, switch allocation is prioritized for VC 0 as long as it is occupied. Once the tail flit of the privileged packet arrives, the local ingress/egress port to the core is unblocked.

To accommodate for the case where multiple privileged packets arrive at the same router, each router is augmented with a counter that counts the number of outstanding privileged packets. When the head flit of a privileged packet reaches the router, the counter is incremented by one, and when the tail flit of a privileged packet reaches the router, the counter is decremented by one. When the counter is not equal to zero, we cut off the connection between the router and its corresponding local port because there are outstanding transfers of privileged packets.

To be compatible with SPEAR mechanism, the head flit and tail flit of a packet should have one extra bit to indicate privilege level. The extra bit can be either encoded into the message, or be calculated using the existing fields if they can be used to deduce the privilege level. For this paper, we assume the extra bit is encoded into the message.

IV. Experimental Setup

The two security mechanisms were implemented both in simulation and hardware. Booksim, a cycle-accurate NoC simulator [11] was used to test the design functionality and to obtain high-level performance results such as packet latency. The designs were then ported to Verilog in order to determine area and delay penalties. We used the RTL code implemented by Becker in his dissertation [12] as the baseline Verilog code.

For BookSim simulation, unless stated otherwise, we simulated a 3x3 mesh network, with number of VCs set to 4 and the buffer size for each VC set to 8. The packet size is randomized from 3 to 6. We used the packet latency as the measurement of performance. The switch allocator type is set to *select* to allow prioritizing switch allocation for VC 0.

We chose *ROUTER_TYPE_VC* as the router type in the Verilog implementation. Again the number of VCs is set to 4 and the buffer size is set to 8 for consistency between the Verilog implementation and BookSim simulation.

V. Results

A. Experimental results for BookSim simulation

We simulated single node and multiple nodes conducting DoS attacks in BookSim. In Figure 5, the flit counter for Node 0 exceeds threshold during epoch 0. As a result, Node 0 is temporally shut down during epoch 1 & 2 and notified to reschedule its injection process. After epoch 3, Node 0 still has a counter number over threshold, which shows that it is truly compromised. Therefore, the network stops receiving packets from node 0 permanently. Figure 6 shows that our mechanism is able to successfully detect and stop multiple nodes conducting DoS attacks. Both Node 0 and Node 1 are stopped from injecting packets into the network for two epochs after their abnormally high injection rates are first detected. Two epochs afterwards, the second phase of detection confirms both Node 0 and Node 1 are truly conducting DoS attacks. As a result, the network stops receiving packets from both nodes.

Figure 7 demonstrates how DoS attack detection mechanism is able to avoid false positives. During epoch 0, Node 0 has an injection rate above the threshold due to high traffic in the network although it is not conducting DoS attack. The network temporarily cuts off injection from Node 0 and send it a notification to reschedule. Node 0 receives the notification and slows down its injection process. In epoch 3, phase two of the detection mechanism observes a normal injection rate from Node 0. Therefore, Node 0 resumes to inject packets into the network normally afterwards.

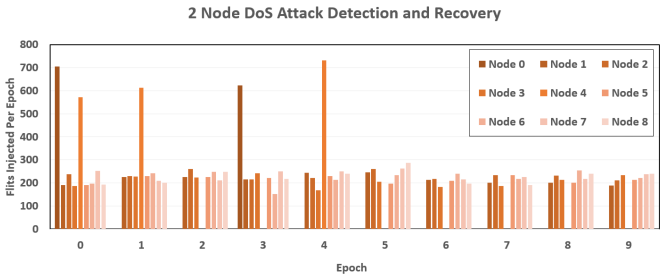

Fig. 6. Node 0 & 1 DoS attack detection and recovery

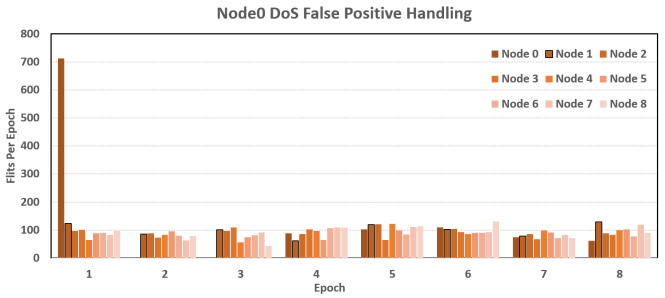

Fig. 7. High traffic at Node 0 but no DoS attack

We also investigated the effect of uncompromised nodes' injection rates on average packet latency in the network. Figure 8 simulates one node conducting DoS attack with other nodes injecting packets at five different rates. The packet size is set to 4 for this case. It is shown that for all normal injections rates the network latency increases significantly when compromised node's injection rate reaches 0.33 packets/cycle. We also verified that this inflection point where network latency experiences an exponential increase is also independent of other factors such as number of VCs. Therefore, our method of choosing a static threshold for DoS detection is justified.

We evaluated the performance overhead of SPEAR with different privileged packet injection frequencies under various conditions in 3x3 and 8x8 mesh network configurations. When privileged packet injection frequency is 1%, the packet latency is even smaller than the case with no privileged packets. This is likely due to the fact that privileged packets are routed immediately instead of having to wait for arbitration, leading to a smaller overall latency for all packets. However, as privileged packet injection frequency increases, the routers must block their VCs more frequently, resulting in a slow down of the non-privileged packets, which increases the average packet latency in the network. In real applications, we expect the ratio of privileged packets to be around 5-10%. We tested 20% privileged packet injection frequency as worst-case scenario. The following section explains SPEAR's performance impact when varying different network parameters such as traffic pattern, number of VCs, and packet injection rate.

We ran simulation with different traffic patterns including Uniform, Neighbor, and Hotspot. The results are shown in Figure 9. In 8x8 mesh network, there is no significant performance

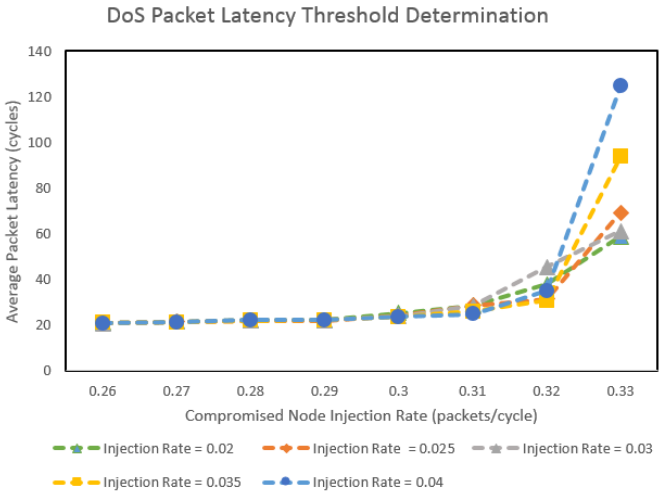

Fig. 8. DoS attack anomalous packet injection rate threshold

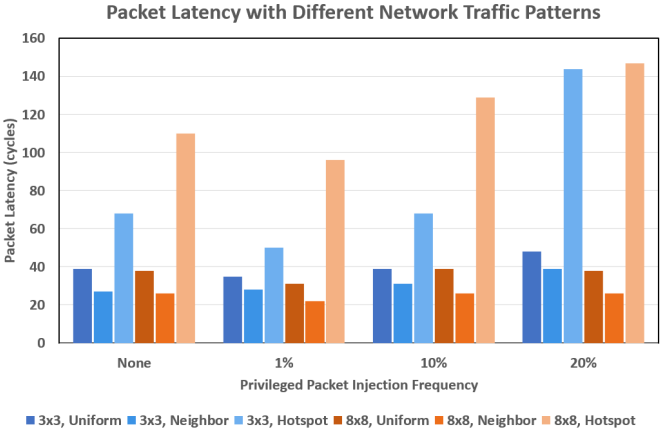

Fig. 9. Packet latency variation with different traffic patterns

overhead for all privileged packet injection frequencies. In 3x3 mesh network, there is only significant performance impact with Hotspot traffic and 20% privileged packet frequency.

Figure 10 shows that performance impact is not significant in 8x8 mesh network with 2, 3 or 4 VCs for all privileged packet injection frequencies. In 3x3 mesh network, performance impact is only noticeable when privileged packet injection frequency reaches 20%.

Lastly, we analyzed performance impact with 0.05, 0.10, 0.15 and 0.16 packets/cycle injection rate. Similar to the previous two figures, Figure 11 shows that performance overhead is negligible in 8x8 mesh network. In 3x3 mesh network, significant performance overhead can only be observed with privileged packet frequency of 20% and injection rate of 0.16 packets/cycle.

B. Hardware overhead

We ported our design to Verilog and synthesized the single router to get the performance and area overhead. The result is given Table I. Since the additional hardware is not on the

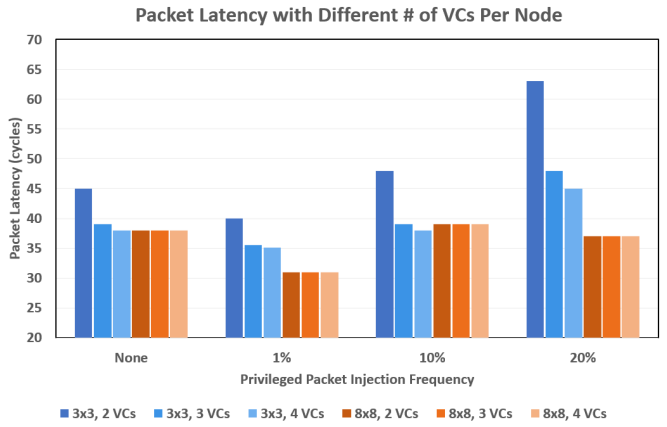

Fig. 10. Packet latency variation with increasing VCs

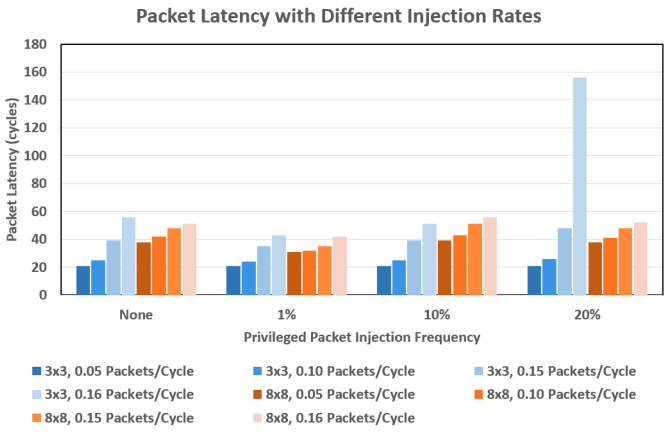

Fig. 11. Packet latency variation with increasing injection rate

critical path of the router, there is no clock speed penalty. The area overhead is also negligible, with only 0.66% for DoS detection mechanism, and 1.1% for both DoS detection and SPEAR.

Our mechanism requires minimal change to the existing architecture of the router. To ensure privileged packets are allocated to VC 0 and non-privileged packets are allocated to other VCs, we only need to add masks bit to the VCs that are eligible during allocation. In other words, privileged packets are only eligible to choose from VC 0, and non-privileged packets are only eligible to choose from VCs other than VC 0. In addition, many router supports priority-based switch allocator. Therefore, we only need to assign VC 0 with a privileged level higher than other VCs to block the allocation of non-privileged packets during the transfer of privileged packets. We do need to add counters to the router for both DoS detection mechanism and SPEAR mechanism, but it does not involve any change to the existing hardware component.

VI. Future Work

In order to improve VC allocation performance, instead of starving non-privileged VCs during secure packet exchange, the SPEAR mechanism can be augmented with a time-division multiplexing scheme. The credit-return policy of the router can be modified to achieve a finer grain control over the local cores packet injection rate. This would improve the efficiency of the DoS recovery mechanism. These are microarchitectural modifications, which, if implemented, will yield performance and QoS improvements.

TABLE I
Hardware overhead

	clock speed slowdown	area overhead
DoS	0	0.66%
DoS + SPEAR	0	1.1%

VII. Conclusion

As the complexity of NoC designs increases, security vulnerabilities arise that need to be addressed. Through large-scale network security is a well-studied topic, little attention has been paid to on-chip networks. In this work, we designed and evaluated mechanisms to protect NoCs from two major forms of attacks, namely DoS and unauthorized secure information extraction. Further, we demonstrated that these mechanisms can be readily implemented with minimal changes to existing NoC infrastructures and with negligible performance and area overheads.

Acknowledgment

The authors would like to thank Doowon Lee and Professor Valeria Bertacco for their exceptional support and guidance in the design and evaluation of this project.

References

[1] S. Anthony, "Intel unveils 72-core x86 knights landing cpu for exascale supercomputing," 2013.
[2] T. Bjerregaard and S. Mahadevan, "A survey of research and practices of network-on-chip," *ACM Computing Surveys (CSUR)*, vol. 38, no. 1, p. 1, 2006.
[3] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrjä, and A. Hemani, "A network on chip architecture and design methodology," in *VLSI, 2002. Proceedings. IEEE Computer Society Annual Symposium on*. IEEE, 2002, pp. 105–112.
[4] S. Evain and J.-P. Diguet, "From noc security analysis to design solutions," in *Signal Processing Systems Design and Implementation, 2005. IEEE Workshop on*. IEEE, 2005, pp. 166–171.
[5] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, and D. Blaauw, "A highly resilient routing algorithm for fault-tolerant nocs," in *Proceedings of the Conference on Design, Automation and Test in Europe*. European Design and Automation Association, 2009, pp. 21–26.
[6] I. Anati, S. Gueron, S. Johnson, and V. Scarlata, "Innovative technology for cpu based attestation and sealing," in *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*, 2013, p. 10.
[7] ARM, "Trustzone," http://www.arm.com/products/processors/technologies/trustzone/, [Online; accessed 14-Dec-2015].
[8] A. G. J. Porquet and C. Schwarz, "Noc-mpu: A secure architecture for flexible co-hosting on shared memory mpsocs." in *Design, Automation Test in Europe Conferene Exhibition (DATE)*, 2011, pp. 1–4.
[9] S. Lokovic and N. Christianos, "Enhancing network-on-chip components to support security of processing elements," in *Proceedings of the 5th Workshop on Embedded Systems Security, WESS '10*, 2010. ACM., pp. 12:1–12:9, New Yorkm NY, USA.
[10] W. J. Dally and B. P. Towles, *Principles and practices of interconnection networks*. Elsevier, 2004.

[11] J. Kim, N. Jiang, D. Becker, G. Michelogiannakis, J. Balfour, B. Towles, and W. Dally, "A detailed and flexible cycle-accurate network-on-chip simulator," 2013.
[12] D. U. Becker, "Efficient microarchitecture for network-on-chip routers," Ph.D. dissertation, Stanford University, 2012.