Securing the Processor-to-Processor and Processor-to-Memory Communication Links

Georgios Angelopoulos, Craig S. Barner and Richard E. Kessler
Marvell Technology Group, Ltd
Marlborough, MA, USA
(georgiosa, cbarner, rkessler)@marvell.com

ABSTRACT

The growing need for privacy and intellectual property protection coupled with the exponentially increasing number of computing devices and generated data obviate for secure compute and storage platforms. However, remote machines with potentially malicious hardware and highly sophisticated software attacks pose a significant challenge. Ensuring confidentiality, integrity and anti-replay of data and code in uniprocessor and (symmetric or distributed shared memory) multiprocessor systems isn't a trivial task. In this work, we review the literature and compare several cryptographic primitives for their performance and computational requirements. Careful selection of the primitives, some of them been considered, to the best of our knowledge, for the first time in the above settings, results in significantly lower area, power and latency overheads.

1 INTRODUCTION

Although security of compute and storage systems has been of utmost importance since the early days of their inception, it still remains an evolving topic that receives significant attention from both the academic and industrial communities. Recent advances in software and hardware design obsolete existing solutions and obviate for security approaches that have never been thought in the past. For instance, in virtualized scenarios with cloud applications being executed on remote, third-party machines located on Infrastructure-as-a-Service (IaaS) premises, not only strong isolation among the numerous applications sharing the resources is desired, but also executed code may not fully trust pieces of the operating system (OS) with higher privileges, such as the hypervisor, of the host system or some of its hardware components. An other example is advances in non-volatile RAM technologies, being able to retain stored values even when removed from power supply for extended periods of time, lowering the barrier for cold-boot and related attacks, compared to rapidly leaking DRAM modules.

Trusted execution environments (TEEs) and isolated application compartments, usually called enclaves, have been developed as a response to the above challenges, claiming that applications' data remain safe even if the OS, hypervisor and hardware have been compromised. During boot-up, a small subset of the chip, called root of trust, initiates a process serially verifying and adding other blocks, gradually expanding the boundaries of the secure part of the chip. Most security vulnerabilities exist when data cross this trust boundary and are exposed to attackers. Usually, the trust boundary is the borders of the chip and two potential avenues for attacks are the processor-to-memory (RAM) and processor-to-processor communication links, as shown in Fig. 1. Inadequate security measures on these links can result in leakage of sensitive information,

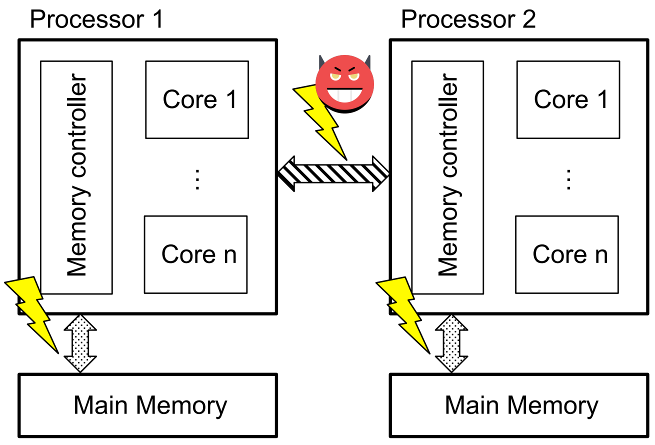

Figure 1: Multiprocessor system with non-authorized user attacking, passively and/or actively, the platform through the off-chip communication links.

e.g. passwords, or even in system-wide attacks. Confidentiality, integrity and freshness of data for intra- and inter-socket communication should generally be of equal importance.

Several encryption and authentication algorithms have been proposed in the literature for this purpose; AES in CBC mode [15] and Merkle hash trees [2, 11] are the most frequent techniques to ensure confidentiality and integrity, respectively. A few alternative methods are summarized in [4, 8, 13] and in their references. However, we are far from a fully secure system as proved by the many recent attacks [5, 14] on platforms initially designed and considered as 'bullet-proof' from a security perspective [6].

Majority of vulnerabilities are architectural/micro-architectural flaws and races leaking secrets, not adhering to known-good coding practices and poor entropy or pseudo-random sources. Cryptographic primitives are less often the culprit of security attacks however they typically exhibit high computational complexities, translated to large die area and power consumption. Because of the speed-gap between processors and memory (processor speed grows by, on average, 18% per year more than memory speed [7]) and the capacitance-dominated off-chip electrical links, adding encryption and authentication latency to these already strained interfaces can be a system performance limiting factor.

For the previously mentioned reasons, carefully choosing the cryptographic primitives of optimal complexity for every use case might enable architects and designers to allocate more resources on architecturally securing the operation of a system, ultimately resulting in smaller attack surface. In this work, we review the

Cipher	Area (kGE)	Latency (cycles)	Normalized Power
AES	78	20	23
PRINCE	4.5	5	1

Table 1: Comparison of block ciphers. Industry standard PnR tools were used and a target frequency of 2.6GHz.

state-of-the-art encryption ciphers and authentication algorithms, and, by carefully employing the right primitives, we demonstrate significant power and area savings compared to [6, 9] and [10, 12] in processor-to-memory and processor-to-processor interfaces.

2 DATA CONFIDENTIALITY, INTEGRITY AND ANTI-REPLAY

Data confidentiality is achieved by concealing data before their off-chip transmission so that only intended recipients with appropriate private keys can decrypt them. This prevents passive attacks such as eavesdropping. Active attacks of corrupting the data with malicious patterns are prevented by verifying that received information creates a signature or cryptographic digest that matches the expectations of a legitimate receiver; this process is called data authentication. Recording a transmitted transaction and inserting it at a later point of time (temporal permutation) or at a different address (spatial permutation) are generally known as replay attacks.

AES is considered the default mechanism for data confidentiality and is the preferred cipher in most intra- and inter-socket systems [6, 9, 15]. Although secure, its high-speed implementation exhibits significant complexities and consumes scarce chip-area resources. We propose the use of an alternative primitive, initially designed for sensor networks, called PRINCE [3]. PRINCE is a 64-bit block cipher with a 128-bit key based on the substitution-permutation principle, achieving security against linear and differential attacks.

The main benefits of using PRINCE cipher compared to AES are the following: *i) Key expansion*: Key schedule is almost instantaneous, without need for storing the expanded key or incurring the expansion overhead in every operation, *ii) Low latency*: Reduced number of rounds, each one exhibiting short logic depth, and *iii) Low area*: Counter mode of operation in block ciphers relaxes the additional introduced latency but short logic depth translates to smaller footprint; utilizing 4-bit SBox'es and balancing their complexity to the linear layer, the required area of PRINCE is significantly reduced, as shown in Table 1 in gate equivalents (GE). In addition, PRINCE encryption and decryption operations are symmetrical, having the α-reflection property. This enables almost the same hardware unit to perform both operations, further reducing area requirements. Above features make PRINCE an ideal candidate against other lightweight ciphers [1].

Counter mode of operation requires a random nonce for every processed block. In our system, we propose the use a counter, combining temporal and spatial information to avoid same-nonce and replay attacks. However, if not carefully performed, managing the counters can result in excessive storage space. In our case, we leverage the already existing protocol information, and achieve minimal overhead. For instance, in the inter-socket link, a function of the sequence numbers of the coherent processor interconnect protocol are used as the counter. Memory authentication is achieved using PRINCE cipher in Galois counter mode (GCM), combined with a parallelizable Merkle tree that balances the additional bandwidth overhead with the on-chip storage requirements. A local cache is also used to further optimize memory accesses and speed up integrity checks.

3 CONCLUSION

Communication buses used to interconnect the processor with the main memory or other processors are usually exposed to passive and active attacks. Securing these high-speed and low-latency links under the tight area and power constraints of modern compute platforms can be challenging. In this work, we review several cryptographic primitives and propose a system that achieves similar security guarantees as state-of-the-art but with significantly lower area and power requirements. This is achieved by selecting appropriate confidentiality and authentication algorithms, and carefully integrating them into the intra- and inter-socket communication interfaces.

REFERENCES

[1] Alex Biryukov and Leo Perrin. 2017. State of the Art in Lightweight Symmetric Cryptography. Cryptology ePrint Archive, Report 2017/511. (2017).

[2] M. Blum, W. Evans, P. Gemmell, S. Kannan, and M. Naor. 1994. Checking the Correctness of Memories. *Algorithmica* 12, 2-3 (Sept. 1994), 225–244. https://doi.org/10.1007/BF01185212

[3] Julia Borghoff et al. 2012. PRINCE - A Low-latency Block Cipher for Pervasive Computing Applications. Cryptology ePrint Archive, Report 2012/529. (2012).

[4] Reouven Elbaz, David Champagne, Catherine Gebotys, Ruby B. Lee, Nachiketh Potlapally, and Lionel Torres. 2009. Transactions on Computational Science IV. Chapter Hardware Mechanisms for Memory Authentication: A Survey of Existing Techniques and Engines, 1–22.

[5] Daniel Gruss, Moritz Lipp, Michael Schwarz, Daniel Genkin, Jonas Juffinger, Sioli O'Connell, Wolfgang Schoechl, and Yuval Yarom. 2017. Another Flip in the Wall of Rowhammer Defenses. *CoRR* abs/1710.00551 (2017).

[6] Shay Gueron. 2016. A Memory Encryption Engine Suitable for General Purpose Processors. Cryptology ePrint Archive, Report 2016/204. (2016).

[7] John L. Hennessy and David A. Patterson. 2006. *Computer Architecture, Fourth Edition: A Quantitative Approach.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[8] Michael Henson and Stephen Taylor. 2014. Memory Encryption: A Survey of Existing Techniques. *ACM Comput. Surv.* 46, 4, Article 53 (March 2014), 26 pages. https://doi.org/10.1145/2566673

[9] D. Kaplan, J. Powell, and T. Woller. 2016. AMD memory encryption (v7). (2016). https://developer.amd.com/wordpress/media/2013/12/AMD_Memory_Encryption_Whitepaper_v7-Public.pdf

[10] Kevin Lepak, Gerry Talbot, Sean White, Noah Beck, S Naffziger, et al. 2017. The next generation amd enterprise server product architecture. *IEEE Hot Chips* 29 (2017).

[11] R. C. Merkle. 1980. Protocols for Public Key Cryptosystems. In *1980 IEEE Symposium on Security and Privacy(SP)*, Vol. 00. 122. https://doi.org/10.1109/SP.1980.10006

[12] David Mulnix. 2017. *Intel Xeon Processor Scalable Family Technical Overview.* Intel Corporation. https://software.intel.com/en-us/articles/intel-xeon-processor-scalable-family-technical-overview

[13] Weidong Shi, Hsien-Hsin S. Lee, Mrinmoy Ghosh, and Chenghuai Lu. 2004. Architectural Support for High Speed Protection of Memory Integrity and Confidentiality in Multiprocessor Systems. In *Proceedings of the 13th International Conference on Parallel Architectures and Compilation Techniques (PACT '04).*

[14] Jo Van Bulck et al. 2018. Foreshadow: Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-order Execution. In *Proceedings of the 27th USENIX Conference on Security Symposium (SEC'18).*

[15] Youtao Zhang, Lan Gao, Jun Yang, Xiangyu Zhang, and Rajiv Gupta. 2005. SENSS: security enhancement to symmetric shared memory multiprocessors. In *11th International Symposium on High-Performance Computer Architecture.* 352–362.