Self-configuration of latency-efficient security enhancements for MPSoC communications monitoring

Pascal Cotret, Guy Gogniat, Jean-Philippe Diguet
Laboratoire Lab-STICC
Université de Bretagne-Sud
Lorient, France
name.surname@univ-ubs.fr

***Abstract*—Nowadays, security is a key constraint in MPSoC development as many critical and secret information can be stored and manipulated within these systems. One strategic point of a bus-based MPSoC is the communication architecture as all data goes through it. Most solutions are currently built at the software level; we believe hardware enhancements also play a major role in system protection. Our approach relies on low complexity distributed security filters connected to all critical IPs of the system. Implementations on a Xilinx xc6vlx240t Virtex-6 FPGA show a latency decrease of 33 % compared to existing efforts while a reconfigurable version of such security services gives a 37% area overhead on a simple dual-processor case study with a 33% latency decrease on a sample image processing application.**

I. Introduction

Embedded systems are facing an increasing number of threats as attacker's motivation is raising every day. High technology devices contain many sensitive information (passwords and confidential contents) that needs to be protected from software and hardware attacks. Reconfigurable technologies such as FPGAs are a good candidate to build such systems as they embed processors, memories and application-specific IPs in a single chip with moderate development costs. When dealing with logical attacks (e.g. targeting the external memory through code/data corruption), main existing solutions are based on software countermeasures. However, relying the system security on software-only solutions may not be adapted for high constrained embedded systems. This paper proposes a solution with reconfigurable hardware security enhancements aiming to protect a bus-based MPSoC from logical attacks while keeping a good area/latency overhead. This paper is organized as follows. Section II gives an overview of the threat model taken into account in this work. Section III presents the overall structure of security enhancements while Section IV shows some results of FPGA implementations. Section V highlights main perspectives.

II. Threat context

Considering an FPGA implementation of a MPSoC, it is assumed that attackers can only tamper with the system using logical attacks (side-channel and other physical threats are not considered). As the target FPGA is considered as trusted, the only way to access the system is through the external memory and the external bus. For many applications, building a flexible solution where only the most critical code/data sections are protected with cryptographic services (instead of ciphering the whole external memory) is a good compromise to keep an acceptable area/latency overhead. In this case, attackers still have possibilities to jeopardize the system by tampering plaintext sections of the external memory. That is why security enhancements have to monitor and detect any abnormal behavior and to propose a solution to reconfigure the system with new parameters to counter the current attack. The key contributions of this work include:

- Demonstration of reconfigurable firewall enhancements.
- Flexible cryptographic services.
- Case study implementations.

III. Hardware firewalls

This work is based on security enhancements embedded in the FPGA chip. It provides a low-latency solution based on hardware firewalls integrated in each IP bus interface providing protection against read/write access and format disruptions (this is done by *Local Firewalls*). The firewall connected to the external memory controller (*Cryptographic Firewall*) adds flexible cryptographic services to protect the memory with confidentiality and/or authentication (Figure 1).

A. Static features

When a data comes from the AXI system bus, it is stored in the *Firewall Interface* while other information (such as address, format and read/write modes) are sent to the *Security Builder*. The *Correspondence Table* indicated the location of the security policy associated with the bus address: security policies contain cryptographic information (such as keys), read/write access and format rules for a given address space. Then, these parameters are sent to the *Checking Module* which compares the system bus parameters with the values extracted from the security policy; at this step, if cryptographic operations are needed, the *Crypto Module* (based on an AES-GCM algorithm) manages the encryption/decryption and authentication tasks with a dedicated BRAM for cryptographic information storage. Once *Checking Module* complete its operations, a *check_out* signal is sent to the *Firewall Interface* to confirm or not the data validity (security policies are verified or not). Finally, *Firewall Interface* provides the final data and manages synchronization tasks in order to fit with the output bus interface. Using this method, the system is protected against logical attacks aiming to tamper the external memory without encrypting the whole memory which would have a strong impact in terms of latency.

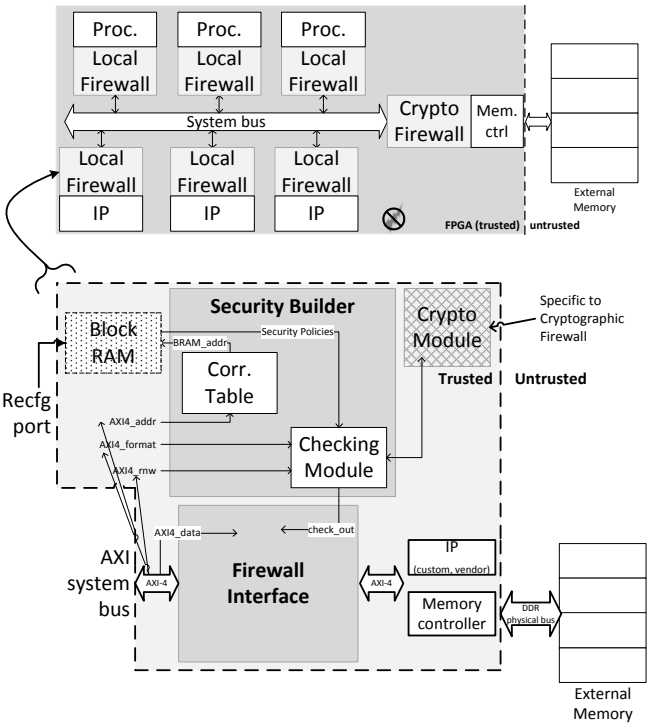

Fig. 1. Firewall-enhanced multiprocessor architecture

B. Attack detection and reconfiguration

When an attack is detected (using the *check_out* signal from the *Checking Module*), BRAM contents must be updated with new security policies in order to keep a safe execution environment for the target MPSoC. A specific architecture is implemented around a dedicated processor aiming to manage reconfiguration and attack reporting tasks. On an attack event, the system must be reconfigured in order to avoid malicious data leakage. One very important issue during firewall reconfiguration is the data availability while switching between two security policies. Using AXI protocol features (especially handshake exchanges), a mechanism is implemented in the *Firewall Interface* to manage runtime reconfiguration.

In most critical cases, if too many attacks are detected, an IP can be put in quarantine or the system can be rebooted with the initial bitstream and security configuration depending on security restrictions defined by the user.

IV. Implementation results

All the following results have been implemented on a ML605 Xilinx board including a Virtex-6 xc6vlx240t FPGA. Measuring area overheads is done comparing firewalls with a simple case study where a MPSoC embeds 2 Microblaze processors, shared Block RAM, an image processing IP and an external memory.

4 Local Firewalls and 1 Cryptographic Firewall are needed for the protection of this case study. Three options are considered: the MPSoC without firewalls, the static firewall-enhanced MPSoC and the reconfigurable version. Area results are summarized in Table I.

The firewall-enhanced case study has a quite high overhead:

	Slices	Regs	LUTs	BRAMs
Unprot. solution	5,446	7,195	8,354	32
Local F.	99	123	293	1
Crypto F.	1,304	2,161	2,689	15
Firewalls w/o recfg	7,302 +34.08%	9,848 +36.87%	12,215 +46.22%	51 +37.25%
Recfg protection	7,442 +36.65%	9,913 +37.78%	12,405 +48.49%	51 +37.25%

TABLE I
Table of standalone results

this is mainly due to the cryptographic module embedded in the firewall attached to the external memory controller. The logic added for reconfiguration purposes implies a quite high area overhead of around 40% compared to the static firewall implementation.

In terms of latency, a centralized approach like SECA would give a 6.27% latency overhead on a sample image processing application while our solution has a 4.18% overhead: it represents a 33% latency decrease.

V. Conclusion and perspectives

This work presents reconfigurable hardware firewall enhancements for bus-based MPSoCs. These firewalls protect memories and memory-mapped IPs according to user-defined security policies. It allows developers to get runtime reconfiguration with a low area overhead compared to a static solution. This work corresponds to a trade-off between [1] and [3] with an additional reconfiguration feature which is not implemented in other bus-based solutions. Security parameters are defined using memory mapping domain, a thread-specific granularity where each thread has its own security policy should be considered in further work.

Acknowledgment

The work presented in this paper was realized in the frame of the SecReSoC project number ANR-09-SEGI-013, supported by a grant of the French National Research Agency (ANR).

References

[1] J. Coburn, S. Ravi, A. Raghunathan, and S. Chakradhar, "SECA: Security-Enhanced Communication Architecture," in *Proc. 2005 Int. Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES)*, Sep. 2005, pp. 78–89.

[2] L. Fiorin, G. Palermo, and C. Silvano, "A monitoring system for NoCs," in *Proc. 3rd Int. Workshop on Network on Chip Architectures (NoCArc)*, Dec. 2010, pp. 25–30.

[3] L. Fiorin, G. Palermo, S. Lukovic, and C. Silvano, "A data protection unit for NoC-based architectures," in *Proc. IEEE/ACM 5th IEEE/ACM Int. Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Sep. 2007, pp. 167–172.

[4] P. Cotret, J. Crenne, G. Gogniat, and J.-P. Diguet, "Bus-based MPSoC security through communication protection: A latency-efficient alternative," in *Proc. IEEE 20th Annual Int. IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, Apr. 2012, pp. 200–207.

[5] J.-P. Diguet, S. Evain, R. Vaslin, G. Gogniat, and E. Juin, "NOC-centric security of reconfigurable SoC," in *Proc. ACM/IEEE 1st Int. Symposium on Network-on-Chips (NOCS)*, May 2007, pp. 223–232.

[6] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, "Security in embedded systems: Design challenges," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, no. 3, pp. 461–491, Aug. 2004.

[7] J. Crenne, R. Gogniat, Guy anand Vaslin, G. Gogniat, J.-P. Diguet, R. Tessier, and D. Unnikrishnan, "Configurable memory security in embedded systems," *ACM Transactions on Embedded Computing Systems (TECS) (accepted/to appear)*, vol. Jan., 2012.