CryptArchi 2017

LABORATOIRE HUBERT CURIEN
UMR • CNRS • 5516 • SAINT-ETIENNE

15th International Workshop
on Cryptographic Architectures
Embedded in Logic Devices

SLOVENSKÁ AKADÉMIA VIED ACADEMIA SCIENTIARUM SLOVACA

Smolenice, Slovakia

June 18th - 21st 2017

http://www.cryptarchi.org

STU FEI
SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA
FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY

cnrs

UNIVERSITÉ JEAN MONNET SAINT-ÉTIENNE

Microsemi

UNIVERSITÉ DE LYON

Committees

Steering committee

Nathalie Bochard	Jean Monnet University, Saint-Etienne, France
Viktor Fischer	Jean Monnet University, Saint-Etienne, France
Otokar Grosek	Slovak University of Technology, Bratislava, Slovakia
Karol Nemoga	Slovak Academy of Sciences, Bratislava, Slovakia

Program committee

Nathalie Bochard	Jean Monnet University, Saint-Etienne, France
Lilian Bossuet	Jean Monnet University, Saint-Etienne, France
Jean-Luc Danger	TELECOM ParisTech, Paris, France
Milos Drutarovsky	Technical University of Kosice, Slovakia
Viktor Fischer	Jean Monnet University, Saint-Etienne, France
Kris Gaj	George Mason University, Fairfax, USA
Guy Gogniat	University of South Brittany, Lorient, France
Otokar Grosek	Slovak University of Technology, Bratislava, Slovakia
Sylvain Guilley	TELECOM ParisTech, Paris, France
Tim Güneysu	Ruhr University Bochum, Germany
Robert Lórencz	Czech Technical University Prague, Czech Republic
Philippe Maurine	CEA-TECH, Gardanne, France
Nele Mentens	KU Leuven, ESAT, COSIC, Belgium
Karol Nemoga	Slovak Academy of Sciences, Bratislava, Slovakia
Olivier Sentieys	University of Rennes I, France
Lionel Torres	University of Montpellier II, France
Ingrid Verbauwhede	KU Leuven, ESAT, COSIC, Belgium

Participants

Wael Adi	Technical University of Braunschweig, Braunschweig, Germany
Alain Aubert	Lab. Hubert Curien, Université de St-Etienne, France
Jan Belohoubek	Czech Technical University, Prague, Czech republic
Florent Bernard	Lab. Hubert Curien, Université de St-Etienne, France
Nathalie Bochard	Lab. Hubert Curien, Université de St-Etienne, France
Brice Colombier	Lab. Hubert Curien, Université de St-Etienne, France
Jean-Luc Danger	Telecom ParisTech, France
Markus Dichtl	Siemens AG, Germany
Stanilav Jerabek	Czech Technical University, Prague, Czech republic
Tomas Fabsic	Slovak University of Technology, Bratislava, Slovakia
Oswald Farin	DGA-MI, Rennes, France
Viktor Fischer	Lab. Hubert Curien, Université de St-Etienne, France
Kris Gaj	George Mason University, Fairfax, USA
Ondrey Gallo	Slovak University of Technology, Bratislava, Slovakia
Otokar Grosek	Slovak University of Technology, Bratislava, Slovakia
Viliam Hromada	Slovak University of Technology, Bratislava, Slovakia
Stanislav Jerabek	Czech Technical University, Prague, Czech republic
Khaled Karray	Telecom ParisTech, France
Marcel Kleja	MICRONIC, Slovak Republic
Filip Kodytek	Czech Technical University, Prague, Czech republic
Marek Laban	MICRONIC, Slovak Republic
Robert Lorencz	Czech Technical University, Prague, Czech republic
Audrey Lucas	CNRS, IRISA, Rennes, France
Ayoub Mars	Technical University of Braunschweig, Braunschweig, Germany
Vojtech Miskovsky	Czech Technical University, Prague, Czech republic
Ugo Mureddu	Lab. Hubert Curien, Université de St-Etienne, France
Karol Nemoga	Slovak Academy of Sciences, Bratislava, Slovakia
Richard Newell	System-on-Chip Products Group, Microsemi Corp., USA
Elie Noumon Allini	Lab. Hubert Curien, Université de St-Etienne, France
Martin Novotny	Czech Technical University, Prague, Czech republic
Oto Petura	Lab. Hubert Curien, Université de St-Etienne, France
Francesco Regazzoni	ALaRI - USI, Lugano, Switzerland
Tania Richmond	Laboratoire IMATH, Toulon, France
Torsten Schuetze	Rohde & Schwarz SIT GmbH, Stuttgart, Germany
Arnaud Tisserand	CNRS, Lab-STICC, Lorient, France
Felipe Valencia	ALaRI - USI, Lugano, Switzerland
Michal Varchola	ELIT SYSTEMS, s.r.o., Kosice, Slovakia

MONDAY June 19th 2017

9 :00 – 10 :30 **Session I: Design protection** **Chair: V. Fischer**

30 min 01-**Audrey Lucas**, Arnaud Tisserand, CNRS - IRISA - LabSTICC, France
ECC Protections against both Observation and Pertubation Attacks **P 17**

30 min 02-**Brice Colombier**, Lilian Bossuet, David Hély, Lab. Hubert Curien, St-Etienne
Centrality Indicators For Efficient And Scalable Logic Masking **P 29**

30 min 03-**Tania Richmond**, Laboratoire IMATH, Toulon, France
Low-complexity DPA Countermeasure for Resource-Constrained Embedded McEliece Implementation **P 35**

11 :00 – 12 :00 **Session II: Side channel attacks** **Chair: M. Dichtl**

30 min 04-**Vojtech Miskovsky**, Czech Technical University, Prague
Influence of Fault Tolerant Design Techniques on Resistance against Differential Power Analysis **P 42**

30 min 05-**Jan Belohoubek**, Czech Technical University, Prague
The Design-Time Side-Channel Information Leakage Estimation **P 50**

14 :30 – 15 :30 **Session III: Physical Unclonable Functions** **Chair: F. Bernard**

30 min 06-**J-Luc Danger**, O. Rioul, S.Guilley, A. Schaub, Télécom ParisTech
Formalism to assess the Loop-PUF entropy and reliability **P 59**

30 min 07-**W. Adi**, A. Mars, S. Mulhem, Technical University of Braunschweig, Germany
Clone-resistant structures in Microsemi SoC units **P 66**

16 :00 – 17 :00 **Session IV: Implementation of cryptographic architectures with demo** **Chair: V. Fischer**

30 min 08-**Marcel Kleja, Marek Laban**, Viktor Fischer, Technical University of Kosice, Micronic, Slovakia
Secure Portable USB Data Storage **P 77**

30 min 09-**Brice Colombier** et al., Lab. Hubert Curien, St-Etienne
Complete activation scheme for IP design protection (demo) **P 83**

TUESDAY June 20th 2017

9 :00 – 10 :30 **Session V: Random Number Generation** **Chair: J.L. Danger**

15 min 10-**Markus Dichtl**, Siemens Corporate Technology, Germany
ALESSIO, a Research Project on Updatable Security Components of Persistent Industrial Embedded Systems **P 84**

30 min 11-**Markus Dichtl**, Siemens Corporate Technology, Germany
Fibonacci Ring Oscillators as True Random Number Generators - A Security Risk **P 86**

15 min 12-**Viktor Fischer**, LaHC St-Etienne, France
Design and evaluation of a physical random number generator (guideline for certification) **P 94**

30 min 13-**Elie Noumon Allini**, Florent Bernard, Viktor Fischer, LaHC St-Etienne, France
An illustration of a new certiffication approach for TRNGs **P 103**

11 :00 – 12 :00 **Session VI: Attack schemes** **Chair: M. Novotny**

30 min 14-**K. Karray**, J-L Danger, S. Guilley, A. El Aabid, Télécom ParisTech
Attack tree construction: an application to the connected vehicle **P 118**

30 min 15-Tomas Fabsic, **Ondrej Gallo, Viliam Hromada**, Slovak University of Technology, Bratislava, Slovakia
Demonstration of the Acoustic Cryptanalysis **P 129**

14 :30 – 15 :30 **Session VII: Post Quantum Cryptosystems** **Chair: A. Tisserand**

30 min 16-Francesco Regazzoni, **Felipe Valencia**, ALaRI - USI, Lugano, Switzerland
Power Analysis Resistance of Lattice-based Cryptosystems

30 min 17-Malik Umar Sharif, Ahmed Ferozpuri and **Kris Gaj**, George Mason University, USA
Lessons Learned from High-Speed Implementation and Benchmarking of Two Post-Quantum Public-Key Cryptosystems **P 136**

16 :00 – 17 :00 **Session VIII: Implementation of cryptographic architectures** **Chair: K. Gaj**

30 min 18-Gabriel Gallin, **Arnaud Tisserand**, CNRS - IRISA - LabSTICC, France
Hardware Architectures for HECC **P 144**

30 min 19-**Richard Newell**, Microsemi Corp., USA
Improving Trust in the FPGA Supply Chain using Blockchain and Keyless-Signature Technology **P 156**

ABSTRACTS

***Audrey Lucas**, Arnaud Tisserand, CNRS - IRISA - LabSTICC, France*

Among physical attacks, two types are considered as important threats for embedded cryptoprocessors: observation attacks [1] (or side channel attacks) and perturbation attacks [2] (or fault injection attacks). The first ones use external measurements of the circuit execution to guess secrets (e.g. timings, power consumption, electromagnetic radiation). The second ones disrupt the circuit and exploit its unspecified behavior, directly or not, to deduce secrets. In most of state of the art works, countermeasures only protect the crypto-processor from one type of attacks (i.e. side channel or fault attacks). However, in many cases the crypto-system stays or becomes vulnerable to the other type of attacks.

In this work, we focus on protections against both types of attacks simultaneously for scalar multiplication in elliptic curve cryptography (ECC). This is the main operation: k x P where k is a scalar (the secret key in some primitives) and P a public curve point. Scalar multiplication is vulnerable to observation attacks when using weak algorithms where point addition and point doubling operations have different cost or behavior [3]. One countermeasure consists in using double and add always algorithm [4]. Unfortunately, this algorithm is weak to fault attacks (injecting a fault during a dummy point addition does not impact the final result, revealing that the target operation was a dummy one). A common protection against fault attacks on ECC is to verify if the current point is on the curve (by applying its coordinates into the curve equation) [5]. Verification of the current point detects an attack at a specific time during scalar multiplication. For instance, one can choose to only verify the final point of the scalar multiplication for a very low overhead (2 field multiplications, 4 field additions and 1 field multiplication by a scalar). But the secret key may leak before the end. More frequent verification is possible to detect attacks as soon as possible. For instance, one can verify the result after the systematic point doubling in each iteration. This leads to regular scalar multiplication for some coordinate types.

Unlike the current point, the scalar k is not protected by this verification method. To alleviate this vulnerability, we developed a new countermeasure to protect k. Our aim is to count that at the ith iteration, there is a point addition or not. Our protection is efficient against fault attacks named bit flips. The protection cost is about $\log2(k)/2$ integer additions for one scalar multiplication. We are working on ways to perform this verification without leakage to avoid observation attacks. We combined and tested these two countermeasures on different coordinates types for Weierstrass curves. Table 1 reports the computation time overheads in the worst case (i.e., a verification after each point doubling). The cost of this type of regular protection is quite small, but it can be reduced if one choose to only verify the point coordinates less frequently (e.g., w-NAF algorithms).

Although the scalar protection is very cheap, it does not protect against stuck-at faults on k digits. We are working on protections against this type of fault without observation leakage.

Algorithms	Coordinates		
	Affine	Jacobian	Projective
Double and add	5.7%	17.2%	9.8%
Montgomery ladder	6.0%	18.0%	8.8%
NAF	6.1%	16.9%	11.2%
w-NAF	7.2%	21.2%	12.5%

Table 1: Computation overheads in the worst case for Weierstrass curves.

Acknowledgments

This work is partly funded by DGA-PEC and the HAH project (Labex CominLab and Lebesgue, Brittany Region, http://h-a-h.inria.fr/).

References

[1] S. Mangard, E. Oswald, and T. Popp. Power analysis attacks - revealing the secrets of smart cards. Springer, 2007.

[2] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan. The sorcerer's apprentice guide to fault attacks. Proceedings of the IEEE, 94(2):370–382, February 2006.

[3] Eric Brier and Marc Joye. Weierstraß Elliptic Curves and Side-Channel Attacks. In Proc. Public Key Cryptography - PKC, pages 335–345, Paris, France, 2002.
[4] Jean-Sébastien Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In Proc. Cryptographic Hardware and Embedded Systems- CHES, pages 292–302, Worcester, MA, USA, 1999.
[5] I. Biehl, B. Meyer, and V. Müller. Differential Fault Attacks on Elliptic Curve Cryptosystems. In Proc. Advances in Cryptology - CRYPTO, pages 131–146, Santa Barbara, California, USA, 2000.

02-Centrality Indicators For Efficient And Scalable Logic Masking **P 29**

***Brice Colombier**, Lilian Bossuet, David Hély, Lab. Hubert Curien, St-Etienne, Univ. Grenoble Alpes, LCIS, Valence*

Modifying the logic at register transfer level can help to protect a circuit against counterfeiting or illegal copying. By adding extra gates, the outputs can be controllably corrupted. Then the circuit operates correctly only if the right value is applied to the extra gates. The main challenge is to select the best position for these gates, to alter the circuit's behaviour as much as possible. However, another major point is the computational efficiency of the selection process, which should be as good as possible for integration in EDA tools. State-of-the art methods, based on fault analysis, are very demanding and cannot cope with large netlists in a reasonable runtime. We propose to use centrality indicators instead. Centrality is used to identify the most significant vertices of a graph. We show that, when used to select the nodes to modify, they lead to low correlation between original and altered outputs while being computationally efficient. We give experimental results on combinational benchmarks and compare to other previously proposed heuristics. We show that this method is the only efficient selection heuristic that is able to handle large netlists and integrate smoothly into EDA tools.

03-Low-complexity DPA Countermeasure for Resource-Constrained Embedded McEliece Implementation **P 35**

***Tania Richmond**, Martin Petrvalsky, Milos Drutarovsky, Pierre-Louis Cayrel, Viktor Fischer, Laboratoire IMATH, Toulon, France, Technical University of Kosice, Slovakia, Lab. Hubert Curien, St-Etienne*

In this paper, we present a differential power analysis attack on the McEliece public-key cryptosystem. We demonstrate that a part of a private key – permutation matrix – can be recovered using the power analysis. We attack a software implementation of a ‘secure’ permutation that was proposed by Strenzke et al at PQCrypto 2008. The cryptosystem is implemented on a 32-bit ARM based microcontroller and power consumption measurements of the device provide us leakage. In addition, we outline a novel countermeasure against the introduced attack. The countermeasure uses properties of linear codes and does not require large amount of random bits which can be profitable for low-cost embedded devices.

04- Influence of Fault Tolerant Design Techniques on Resistance against Differential Power Analysis **P 42**

***Vojtech Miskovsky**, Czech Technical University, Prague*

The security is becoming more and more important issue these days, but we still should consider reliability. When we design a cryptographic device e.g. for some mission-critical or another reliability demanding system, we need to make the device not only attack resistant, but also fault tolerant.
In context of the digital design we usually talk about resistance against side channel attacks, because these are not based on cryptographic properties of the cipher, but on properties of its physical implementation. For example power attacks use powertrace of the device to reveal some secret information about the device, usually a cipher key during an encryption.

We know many digital design techniques used to secure the device against power attacks with some area and time overhead. But what happens, if we want to make the device also fault tolerant? Many fault tolerant architectures are based on some kind of redundancy. This redundancy introduces large area and also power consumption overhead. But does this overhead have any influence on the attack resistance? And what about the overhead? Isn't there any way to decrease the overhead by some combined attack resistant and fault tolerant architecture? We try to answer these questions in our research.
We compared vulnerability to differential power analysis of a simple AES cipher implementation and its multiple fault tolerant variants implemented in FPGA. Results of this comparison will be presented.
When we were experimenting with differential power analysis against FPGAs we found out that computation requirements of the attack demand an implementation that is more efficient than scripts in Wolfram Mathematica or MATLAB. We decided to program a high performance and numerically stable application for this purpose. This application and its properties will also be presented.

05-The Design-Time Side-Channel Information Leakage Estimation P 50

***Jan Belohoubek**, Czech Technical University, Prague*

The design of today's circuits is an iterative process, where every iteration could influence the information leakage into the side channel. During the design time, bugfixes must be incorporated, and sometimes even architectural changes are performed.
The principles of a new analytical simulation-based method allowing an efficient side-channel information leakage evaluation in various steps of the digital design flow will be presented. If applied, the method allows to decide, if a certain design decision has a positive or negative influence on the sidechannel information leakage independently of any current or future attack schemes.
The experiments with the benchmark circuits showed, that (when the manufacturing variations are supressed) the NModular-Redundancy offers no additional information leakage compared to the single module and the unbalanced dual-rail implementation offers additional information compared to the single-rail implementation.

Acknowledgement
Computational resources were provided by the CESNET LM2015042 and the CERIT Scientific Cloud LM2015085, provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures".

Reference
[1] C. Y. Chu, Improved Models for Switch Level Simulation. Stanford University, 1988.
[2] J. Schmidt and P. Fišer, "A prudent approach to benchmark collection," in Proc. of 12th Int. Workshop on Boolean Problems (IWSBP), Freiberg (Germany), 2016.
[3] D. Karaklajic, J.-M. Schmidt, and I. Verbauwhede, "Hardware designer's guide to fault attacks," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 21, no. 12, pp. 2295–2306, 2013.

06-Formalism to assess the Loop-PUF entropy and reliability P 59

***J-Luc Danger**, O. Rioul, S.Guilley, A. Schaub, Télécom ParisTech*

Many Methods exist for Physically Unclonable Function in order to enhance their reliability, but they are at the cost of extra hardware, either coming from the PUF core itself, or from the correction method, as fuzzy extraction [5]. Another issue is to better know the PUF entropy, especially when the PUF is used as a cryptographic key generator. The SRAM and Ring Oscillator PUF provide relatively high entropy, as shown experimentally in [4]. The min-entropy is generally better assessed, as studied in Delvaux et al. [2] which presents the min-entropy of strong PUFs. For strong PUFs, it appears that the number of required challenges to get the min-entropy is significantly higher, thus

involving compression and extra complexity to generate key bits. We show in this study that it is possible to obtain a strong PUF, relying on a Loop PUF, which presents a high ratio between reliability and complexity. Moreover the entropy can be known when choosing the minimal and optimal number of challenges, namely Hadamard Codes. As the presence of noise inevitably involves an entropy decrease, this paper presents a method allowing the PUF to compensate this loss to meet the requires entropy. The single Ring-Oscillator of the Loop PUF, the hard-coded challenges and the lightweight error correction provide a low complexity PUF, around 1000 μm^2 , with an access time of 20 ms to get a 64 key bits with a Bit Error Rate (BER) which can be less than 10^{-9} by filltering unreliable bits. As the challenges and responses remain local, the modeling attacks are significantly reduced and are not considered in this work.

07-Clone-resistant structures in Microsemi SoC units **P 66**

W. Adi, *A. Mars, S. Mulhem, Technical University of Braunschweig, Germany*

The concept of Secret Unknown Ciphers SUC is presented. SUC concept appears to be a strange one when proposing to use ciphers which nobody knows. The strongest practical secret is the one which nobody knows. Secret Unknown Ciphers SUCs were introduced by the author in 2009. SUCs are self-created, highly unpredictable unknown-ciphers accommodated in digital non-volatile self-reconfiguring VLSI units. Such VLSI units for accommodating SUCs are still not available. We postulate however, that such VLSI infrastructure may become available in the near future emerging technologies. Realizing such ciphers is a challenging task requiring a “smart software GENNI” to create such ciphers within a short time and disappear out of SoC unit. A research group at the technical university of Braunschweig attained first promising basic results for efficient realization of such ciphers targeting Microsemi SmartFunsion FPGA SoC units or any similar technology. The presentation demonstrates some basic generic protocols for using such SUC digital structures as clone-resistant modules. A broad spectrum of applications in consumer and vehicular electronics is expected when using such ciphers resulting with relatively low-cost, provable, clone-resistant or even possibly unclonable units. The technique allows manufacturer-independent personalization/security in SoC units having non-volatile technology. This allows, commercially-efficient, FPGA units, by creating tiny SUC module among the functional core structures to protect intellectual property rights in such units and create unclonable physical electronic units. The concept makes break-one break-all attacks infeasible on such systems. Each unit requires to be attacked individually and thus frustrates attackers by making legal units always cheaper than cloning them resulting with “pragmatic security”. Few practical use cases and protype implementation samples are also presented.

08-Secure Portable USB Data Storage **P 77**

Marcel Kleja, Marek Laban*, Viktor Fischer, Technical University of Kosice, Micronic, Slovakia, Lab. Hubert Curien, St-Etienne, France*

USB flash drive, also called pen drive or USB stick is a popular means of data storage, back-up and transfer. Some cryptographically secure versions of the stick exist, but only few of them can be trusted (if any).
Secure Portable USB Data Storage device was developed as Demonstrator 2 in the framework of the European H2020 project HECTOR (Hardware Enabled CrypTO and Randomness). It is based on a flash based FPGA SoC - Microsemi SmartFusion2 - which features a 32-bit ARM based microcontroller. The device was developed and tested on the HECTOR evaluation platform, which constitutes a modular system and makes evaluation of device features easier by using existing data and control interfaces.
The talk describes developed device and the various challenges during the development process. One of the aims of the talk is also to show the exploitation of the HECTOR project outputs like physically unclonable function, true random number generator and authenticated encryption algorithm in a future commercial device.

09-Complete activation scheme for IP design protection (demo) **P 83**

Brice Colombier, *Ugo Mureddu, Marek Laban, Oto Petura, Lilian Bossuet, Viktor Fischer., Lab. Hubert Curien, St-Etienne, france, MICRONIC, Slovakia*

Intellectual Property (IP) illegal copying is a major threat in today's integrated circuits industry which is massively based on a design-and-reuse paradigm. In order to fight this threat, a designer must track how many times an IP has been instantiated. Moreover, illegal copies of an IP must be unusable. We propose a hardware/software scheme which allows a designer to remotely activate an IP with minimal area overhead. The software modifies the IP efficiently and can handle very large netlists. Unique identification of hardware instances is achieved by integrating a TERO-PUF along with a lightweight key reconciliation module. A cryptographic core guarantees security and triggers a logic locking/masking module which makes the IP unusable unless the correct encrypted activation word is applied. The hardware side is implemented on several FPGAs.

10-ALESSIO, Project on Updatable Security Components of Persistent Industrial Embedded Systems **P 84**

Markus Dichtl*, Siemens Corporate Technology, Germany*

Security is an inevitable precondition for the smart factory. The required security mechanisms can either be implemented in dedicated hardware or in software. As industrial devices tend to be used for longer periods of time than consumer products, and as security requirements are getting stricter in the course of time, the updatability of security components is indispensable for industrial embedded systems.
This need was acknowledged by the German Federal Ministry of Education and Research, which is funding the project ALESSIO investigating solutions for this problem. ALESSIO started in January 2017.
In addition to the evident approach to achieve more demanding security requirements by updating software components, ALESSIO also analyses the usage of modified FPGA configurations to improve security for products in the field. For a secure update of an FPGA, a secure bitstream update mechanism is required. This aspect is investigated in ALESSIO for SOCs containing both hard processors and an FPGA.
The participants in the ALESSSIO project are Infineon Technologies AG (prime), Giesecke & Devrient, Fraunhofer Institute for Applied and Integrated Security, Technical University of Munich, WIBU Systems AG, and Siemens AG.

11-Fibonacci Ring Oscillators as True Random Number Generators - A Security Risk **P 86**

Markus Dichtl*, Siemens Corporate Technology, Germany*

Fibonacci ring oscillators are easily implemented on FPGAs and ASICs and seem to be good source of true randomness. The randomness is assumed to be caused by chaotic oscillations. However, in this paper Fibonacci ring oscillators are shown to have a risk to oscillate periodically instead of chaotically. The security implications of this are discussed. The probability of the occurrence of the periodic oscillations is determined experimentally on an FPGA for Fibonacci ring oscillators of lengths 16 and 32. Means to overcome the problem of the periodic oscillations are also discussed.

12-Design and evaluation of a physical random number generator (guideline for certification) **P 94**

***Viktor Fischer**, LaHC St-Etienne, France*

French DGA (Direction Générale de l'Armement) is responsible for security in high-security cryptographic applications. The French RNG evaluation scheme is based on the German document AIS 20/31. DGA considered that for high-end security applications some additional guideline is necessary to complete AIS 31 (the PTRNG part).
In 2017, David Lubicz edited the document: "Design and evaluation of a physical random number generator integrated in an electronic chip".
The objective of this talk is to present briefly the document, then the following talk (Elie Noumon) will illustrate the DGA document on a PLL TRNG design.

13-An illustration of a new certiffication approach for TRNGs **P 103**

***Elie Noumon Allini**, Florent Bernard, Viktor Fischer, LaHC St-Etienne, France*

Random number generators represent important cryptographic primitives. They generate random numbers or random bit streams that are used at many security levels in various cryptographic schemes, and have to be unpredictable and flawless. Unpredictability is guaranteed the best by using a random physical process in a physical True Random Number Generator (TRNG). However, the design of physical TRNGs is a very challenging one because many aspects related to this kind of generators are not mastered as they should be. A work in progress is aiming to propose an approach to evaluate TRNGs and check if they are safe enough to be used in cryptographic schemes. This approach will be illustrated with the evaluation of a PLL-based TRNG secured by dedicated statistical tests.

14-Attack tree construction: an application to the connected vehicle **P 118**

***K. Karray**, J-L Danger, S. Guilley, A. El Aabid, Télécom ParisTech*

Remote connectivity of today's and future cars have increased their capabilities of autonomy and safety, but also their attack surface, as reported by many research papers. In the automotive domain, the security has a direct impact on the users safety. Thus, the management of risk is becoming the main concern of automotive manufacturers, especially for the future fully connected and autonomous cars. One possible way to quantify the overall risk of a system is the systematic construction of attack graphs and attack trees. These formalism are presented as one of the possible solutions in the new Cybersecurity Guidebook for Cyber-Physical Vehicle Systems (SAE-J3061). In this paper we propose to use a graph transformation to formally model the car architecture and its state evolution in order to study possible attack paths. The resulting attack trees are then used to estimate the overall risk of the system. Consequently, it becomes possible to study improvements to build a more secure architecture. The proposed method is designed to support the conceptual phase of the vehicle's cyber-physical system. We illustrate the method on a small example to show how it is possible to prove its efficiency.

15-Demonstration of the Acoustic Cryptanalysis **P 129**

*Tomas Fabsic, **Ondrej Gallo**, **Viliam Hromada**, Slovak University of Technology, Bratislava, Slovakia*

In 2014, an acoustic cryptanalysis attack on RSA was presented by Daniel Genkin, Adi Shamir and Eran Tromer. The authors demonstrated that acoustic emanations from a laptop can be used to reveal the private key in implementations of RSA in older versions of GnuPG. We repeated their attack and in this presentation we state our observations and demonstrate the fundamental principles of the attack.

16-Power Analysis Resistance of Lattice-based Cryptosystems

*Francesco Regazzoni, **Felipe Valencia**, ALaRI - USI, Lugano, Switzerland*

The security of cryptographic schemes is based in the difficulty of solving specific mathematical problems. Also, the nature of the basement problems drives some features of the cryptographic primitive implementations, for instance, memory and execution time. Until today, the most well established schemes are based on integer factorization and discrete logarithm problems. This trend is changing because these problems can be solved in polynomial time by quantum computers. Cryptographic schemes based on lattice problems (Lattice-based cryptography) standout because they can not be solved efficiently by quantum computers and its performance is comparable with current cryptosystems (i.e. RSA).

Despite the theoretic security of any cryptographic scheme, for the implementation, it is necessary to take into account physical attacks, which are attacks that take advantage of the implementation vulnerabilities, overpassing the mathematical hardness. These attacks can recover the secret information exploiting the correlation of physical variables - such as power, time and electromagnetic radiation – with the processed data. To destroy this correlation hiding and masking techniques are applied but it implies an overhead in the resource consumption. In this talk we will summarize state of the art protection against power analysis for lattice based cryptography and we will highlight potential research directions.

17-Lessons Learned from High-Speed Implementation and Benchmarking of Two Post-Quantum Public-Key Cryptosystems P 136

*Malik Umar Sharif, Ahmed Ferozpuri and **Kris Gaj**, George Mason University, USA*

If a quantum computer with a sufficient number of qubits was ever built, it would easily break all current American federal standards in the area of public-key cryptography, including algorithms protecting the majority of the Internet traffic, such as RSA, Elliptic Curve Cryptography (ECC), Digital Signature Algorithm (DSA), and Diffie-Hellman. All traditional methods of dealing with growing computational capabilities of potential attackers, such as increasing key sizes, would be futile.

In Feb. 2016, American National Institute of Standards and Technology (NIST) has published a draft report and announced its plans of starting the standardization effort in the area of post-quantum cryptography. This effort is likely to last years and result in the entire portfolio of algorithms capable of replacing current public-key cryptography schemes. The initial announcement was followed by the official Call for Proposals and Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms, issued in Dec. 2016. As a part of this standardization process, fair and efficient benchmarking of Post-Quantum Cryptography (PQC) algorithms in hardware and software becomes a necessity.

In this talk, we will discuss our hardware high-speed implementations of two PQC schemes:

1. NTRUEncrypt Short Vector Encryption Scheme (SVES), fully compliant with the IEEE 1363.1 Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices, and
2. Multivariate Rainbow Signature Scheme.

For fair comparison, both implementations follow the same PQC Hardware Application Programming Interface (API), proposed by our group. The development process, was also standardized, and included common intermediate deliverables, such as a) detailed flow diagrams, b) choice of supported parameter sets, c) top-level block diagram, d) lower-level block diagrams, e) parameters of a hardware architecture, f) cycle-based timing analysis, g) Algorithmic State Machine (ASM) charts, h) Register-Transfer Level (RTL) code, i) software-generated test vectors, j) comprehensive testbenches, k) results of synthesis and implementation, l) analysis of results, m) lessons learned.

As such, both designs provide a valuable reference for any future hardware implementers of PQC schemes, which is very important in the context of the upcoming NIST standard candidate evaluation process.

To the best of our knowledge, we have developed the first synthesizable HDL code of the entire NTRUEncrypt SVES scheme, reported in the scientific literature or available commercially. Our implementation supports two representative parameter sets specified in the 2009 IEEE 1363.1 Standard: ees1087ep1 and ees1499ep1, optimized for speed, which provide security levels of 192 and 256 bits, respectively. The corresponding public key sizes are 1495B and 2062B, respectively, and the corresponding private key sizes, 87B and 109B.

Our hardware implementation is functionally equivalent to the open source software implementation of the IEEE P1363.1 standard, developed by Security Innovation, Inc., and has been thoroughly verified using test vectors generated using this implementation. The speed up of our hardware design running on Xilinx Virtex-7 XC7VX485T FPGA vs. software implementation, running on the Cortex A9 ARM Core of Zynq 7020, with the clock frequency of 666.7 MHz, is over 400.

The relative contribution of various operations to the total execution time is substantially different for the hardware and software implementations. In software, Polynomial Multiplication amounts to about 90% of the total execution time. On the other hand, our hardware implementation is seriously limited by the sequential nature of the SHA-256 calculations. As a result, the operations that are most critical are hash based operations of the Blinding Polynomial Generation Method (BPGM) and the Mask Generation Function (MGF), amounting to about 83% of the total execution time for both supported parameter sets. At the same time, the Polynomial Multiplication can be almost completely overlapped with the computations of BPGM through the use of pipelining, and thus has a negligible influence on the execution time.

In order to remove the hash function bottleneck, multiple solutions have been proposed, including an unrolled hash function architecture, as well as a replacement of the SHA-2/SHA-1 hash functions by more hardware friendly SHA-3 or a pseudorandom function based on the pipelined implementation of AES.

Using the similar methodology, we have developed a new high-speed hardware implementation of the multivariate Rainbow signature scheme, based on the earlier work by Tang et al., presented at the PQCrypto 2011 conference. Our implementation targets an 80-bit security level, through the choice of the following variant of the Rainbow Signature Scheme: Rainbow (GF(28), 17, 12, 12). It has a public key size of 19.64kB and private key size of 30.94kB. It is based on the parallel hardware design for the Gauss-Jordan elimination, which is capable of solving an NxN system of linear equations over GF(28) in N clock cycles. In an effort to optimize the design, multiple architectures for the two-input and three-input multipliers over GF(28) have been implemented and comprehensively benchmarked. Additionally, a novel hardware architecture for the so called pivoting operation (a part of the Optimized Gauss-Jordan Elimination) has been developed.

Our hardware implementation is functionally equivalent to the software implementation by Jintai Ding and Dieter Schmidt from University of Cincinnati, and has been comprehensively verified using test vectors generated using this implementation. The result generation and analysis is currently in progress, and will be presented at the workshop. As a next step, the current architecture will be extended to support at least one additional parameter set, with the higher security level.

Eventually, our goal is to compare the hardware implementations of NTRUEncrypt SVES and the Rainbow Signature Scheme at the same security level, using the same API, from the point of view of the execution time, resource utilization, and speed-up vs. software, as well as flexibility and scalability in terms of supporting multiple parameter sets. This project is intended to pave the way for the future comprehensive, fair, and efficient hardware benchmarking of the most promising encryption, signature, and key agreement schemes from each of several major post-quantum public-key cryptosystem families.

18-Hardware Architectures for HECC P 144

*Gabriel Gallin, **Arnaud Tisserand**, CNRS - IRISA - LabSTICC, France*

1. Context

Nowadays, there is an increasing number of applications and systems requiring strong security on small hardware devices. Public-key cryptography (PKC) is mandatory for providing key exchange and digital signature. The first

standard for public-key cryptosystems was RSA. However, to be compliant with the current recommended theoretical security levels, RSA based cryptosystems must use large keys – at least two thousand bits – which make them too costly for embedded applications.
Curves based cryptography such as Elliptic Curve Cryptography (ECC) or Hyper-Elliptic Curve Cryptography (HECC) is known to provide a given security level at a lower cost than RSA. For instance, 226-bit ECC keys offer the same security level as 2048-bit RSA. Due to its reduced cost and better performance, ECC is now recommended as the PKC standard.

2. Hyper Elliptic Curve Cryptography

Recent research has pointed out HECC as an attractive alternative to ECC. HECC is based on a different kind of curves, which allows the size of the field elements to be halved, but at the expense of an increased number of finite field operations. In [9], Renes et al. very recently presented software implementations of key exchange and signature schemes based on HECC and Kummer surfaces, targeting embedded processors (ARM Cortex M0 and AVR ATmega). The provided results show very interesting speedups compared to state-of-the-art ECC: 30% speedup for Diffie-Hellman key exchange and up to 70% for signature.
Operations on HECC involve more operations on the underlying finite field than ECC. However, one can observe that in ECC, most of the computations are dependent and must be mostly done in a sequential way. For this reason, the internal parallelism in ECC is quite limited compared to HECC. For instance, in the formulas presented in [9], one can find regular patterns of four to eight independent modular multiplications – the most costly and common finite field operation – feasible in parallel during the whole scalar multiplication. HECC internal parallelism brings forward numerous questions for hardware implementation. Those questions can be summarized as follows: how can one take advantage of the parallelism in HECC to design efficient hardware cryptosystems?

3. Arithmetic Units

In order to build an efficient accelerator, the first step is to build efficient arithmetic units. These units are dedicated to the computations over finite field elements. The most common and costly finite field operation in (H)ECC is the modular multiplication. For instance, depending on the multiplier area, one multiplication requires from 30 to 100 clock cycles depending on the field elements width. In [7], Peter L. Montgomery presented an algorithm for modular multiplication, which is still now the base of state-of-the-art multiplication in prime finite fields. It is known as Montgomery modular multiplication (MMM). Many algorithms have been derived from this paper. They mainly aim at improving efficiency by interleaving the multiplication and modular reduction steps in order to reduce the size of the intermediate data and to gain some speedup. One of the most famous variant is the Coarsely Integrated Operand Scanning (CIOS) method presented by Koc¸ et al. in [5]. However, besides these improvements, Montgomery multiplication still suffers from strong dependencies inside the main loop of partial products accumulation and reduction. This makes hardware implementations difficult to optimize in the case of FPGAs using DSP blocks. In order to reach high frequencies, DSP blocks must indeed use three to four internal pipeline stages. Due to data dependencies, one cannot feed efficiently this pipeline, resulting in a loss of efficiency in the circuit utilization.
In [6], Ma et al. proposed a FPGA implementation of MMM based on an improvement of the algorithm presented by H. Orup in [8]. This implementation is known to be one of the fastest implementations on FPGA in the literature. However, to get rid of some of the internal dependencies, the method implies huge overheads in terms of the size of the computed data, increasing the circuit area of the design.

4. Proposed HECC Architectures

Our research group has been studying arithmetic operators and implementations of hardware accelerators for ECC, with robustness against physical attacks such as Side Channel Analysis (SCA) and faults injections. We are now designing hardware accelerators for HECC scalar multiplication by exploring different types of architectures.
For this, we first improved the hardware utilization of the multiplier unit. We decided to use the classical CIOS method as a basis to design an hyper-threaded modular multiplier. The idea behind this hyper-threaded multiplier is to fill the unused stages of the DSP blocks with other independent modular multiplications. In our multiplier, we enter 3 independent sets of operands f(A1;B1); (A2;B2); (A3;B3)g before the first product P1 = A1 B1 is computed.

This way, all the stages in the DSP blocks are full after the very first latency. This is more efficient for HECC since the internal parallelism allows more than 3 independent multiplications at each step during the scalar multiplication.
We then developed a specific CABA (Cycle Accurate, Bit Accurate) simulator for our architectures. With this simulator, we can study the impact of the type, number and size of the arithmetic units and of the choice between different types of parallel architectures on the performances, circuit area and resistance against physical attacks.
We will present implementation results on different FPGAs for various configurations of our modular multiplier, both in terms of computation time and circuit area. As an example, for 128-bit field elements, we reach the same computation time with half the number of DSP blocks compared to the best state-of-the-art [6]. We will show that it provides a better computation time / circuit area cost trade-off when several modular multiplications can be computed in parallel, which is always the case in HECC.
In a second time, and using our simulator, we will explore various architectures for our accelerator, starting from a classical Harvard architecture and changing architectural parameters, such as the numbers and types of arithmetic units. We will also compare different ways to manage internal data transfers and different control flow implementations. The most interesting configurations will be implemented on FPGA and evaluated on our attack setup.

Acknowledgments
This work is partly funded by the HAH project (Labex CominLab and Lebesgue, Brittany Region, http://h-a-h.inria.fr/).

References
[1] J. W. Bos, C. Costello, H. Hisil, and K. Lauter. Fast cryptography in genus 2. Journal of Cryptology, 28–60, 2016.
[2] H. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, and F. Vercauteren. Handbook of Elliptic and Hyperelliptic Curve Cryptography. Discrete Mathematics and Its Applications. Chapman & Hall/CRC, July 2005.
[3] P. Gaudry. Fast genus 2 arithmetic based on theta functions. Journal of Mathematical Cryptology, 243–265, 2007.
[4] D. Hankerson, A. Menezes, and S. Vanstone. Guide to Elliptic Curve Cryptography. Springer, 2004.
[5] C¸ . K. Koc¸, T. Acar, and B. S. Kaliski, Jr. Analyzing and comparing Montgomery multiplication algorithms. Micro, IEEE, 16(3):26–33, June 1996.
[6] Y. Ma, Z. Liu, W. Pan, and J. Jing. A high-speed elliptic curve cryptographic processor for generic curves over GF(p). In Proc. 20th International Workshop on Selected Areas in Cryptography (SAC), volume 8282 of LNCS, pages 421–437. Springer, Aug. 2013.
[7] P. L. Montgomery. Modular multiplication without trial division. Mathematics of Computation, 519–521, 1985.
[8] H. Orup. Simplifying quotient determination in high-radix modular multiplication. In Proc. 12th Symposium on Computer Arithmetic (ARITH), pages 193–199. IEEE Computer Society, July 1995.
[9] J. Renes, P. Schwabe, B. Smith, and L. Batina. Kummer: Efficient hyperelliptic signatures and key exchange on microcontrollers. In Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES), volume 9813 of LNCS, pages 301–320. Springer, Aug. 2016.

19-Improving Trust in the FPGA Supply Chain using Blockchain and Keyless-Signature Technology P 156

***Richard Newell**, Microsemi Corp., USA*

Blockchain and Keyless Signature Infrastructure (KSI) technologies that only rely on secure message digests and need no secret keys can be used to provide additional assurances that non-volatile FPGA components and systems moving up the supply chain hierarchy from wafer test through to completed systems are trustworthy. This is done by providing cryptographic evidence of the FPGA's provenance using a verifiable time-stamped audit trail of key events in the FPGA life-cycle using blockchain and KSI technology, complementing existing traditional measures. System manufacturers using those FPGAs can keep appending to the extensible information container that represents the entire history of the FPGA (and eventually the system), strengthening trust in the system, preventing counterfeiting at all levels (component and system), and providing a strong verifiable identity for use in the final run-time application.

Presentations

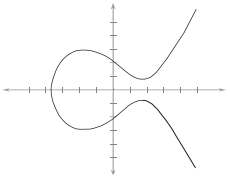
ECC Protections against both Observation and Perturbation Attacks

Audrey LUCAS and Arnaud TISSERAND

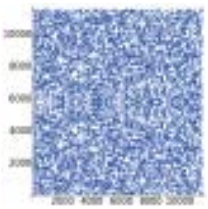
CryptArchi 2017

Outline

Elliptic Curves Cryptography (ECC)



⇓



$$E : y^2 = x^3 + ax + b$$

Scalar multiplication

⇓

$$[k]P = \underbrace{P + P + \ldots + P}_{k \text{ times}}$$

Point *doubling*: DBL

$\neq$

Point *addition*: ADD

Scalar Multiplication Example: Double and Add

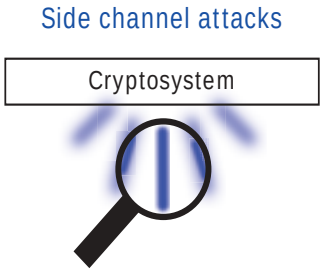
Require: P and $k = (k_{n-1}, \ldots, k_0)_2$.
Ensure: $[k]P$
  $Q \leftarrow \mathcal{O}$
  for $i = n-1$ **to** 0 **do**
    $Q \leftarrow 2 \cdot Q$
    if $k_i = 1$ **then**
      $Q \leftarrow P + Q$
    end if
  end for
  return Q

1		0	0	0	1	
DBL	ADD	DBL	DBL	DBL	DBL	ADD

Physical Attacks

Side Channel Attacks (SCA)

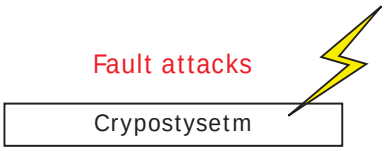
- Computation time, power consumption, ...
- Simple power analysis (SPA), differential power analysis (DPA), ...



Physical Attacks

Fault Attacks (FA)

- Clock, voltage, laser, ...
- Bit flip, stuck-at fault, ...
- Safe error, differential fault analysis (DFA), ...



Physical Attacks

Countermeasures against SCAs

- Randomization: scalar masking, point blinding, scalar recoding, ...
- Uniformization: uniform curve, regular algorithm, ...

Countermeasures against FAs

- Hardware: shielding, sensor, ...
- Redundancy calculation: time, space.
- ECC case:
  - Checking point coordinates at the end of scalar multiplication.

Problem

1		0	0	0	1	
DBL	ADD	DBL	DBL	DBL	DBL	ADD

Double and add

Problem

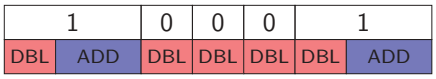
1		0	0	0	1	
DBL	ADD	DBL	DBL	DBL	DBL	ADD

Double and add

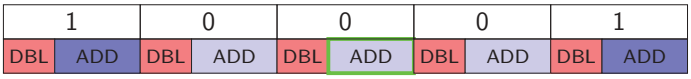
1		0		0		0		1	
DBL	ADD	DBL	ADD	DBL	ADD	DBL	ADD	DBL	ADD

Double and add always

Problem



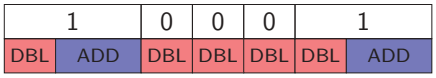
Double and add



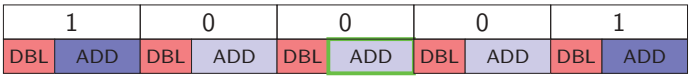
Double and add always - safe error

Problem

Protection for one type of attacks may leave the system vulnerable on other type of attacks.



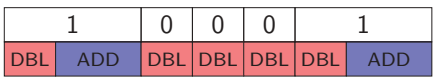
Double and add



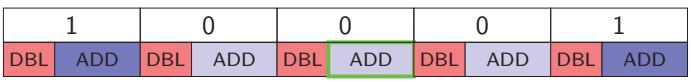
Double and add always - safe error

Problem

Protection for one type of attacks may leave the system vulnerable on other type of attacks.



Double and add



Double and add always - safe error

Protect the system from both types of attacks simultaneously.

Outline

Point Verification

Principle:

- Verify if the current/final point is on the curve [BMM].
- Adaptable for many curves and many coordinates.

Example:

Let $Q = (x, y, z) = [k]P$ a point in projective coordinate of Weierstrass curve. The verification is:

$$y^2z - (x^3 + axz^2 + bz) = 0$$

I. Biehl, B. Meyer, and V. Müller.
Differential Fault Attacks on Elliptic Curve Cryptosystems.
In *Proc. Advances in Cryptology - CRYPTO 2000.*

Verification period:

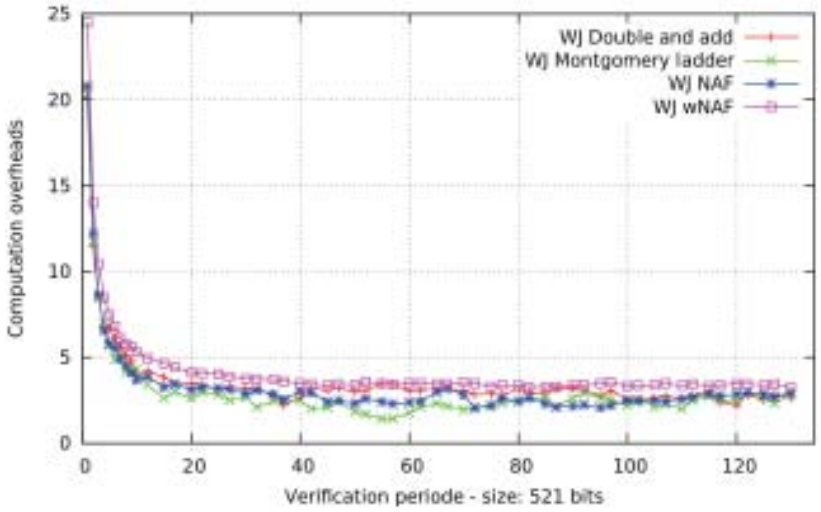
- At the very end: very low cost but late detection.
- Every ℓ iterations: larger cost but earlier detection.
- At each iteration ($\ell = 1$): important cost but immediate detection and possible uniformization of behaviour.

Uniformization:

1			0		0		0		1		
DBL	V	ADD	DBL	V	DBL	V	DBL	V	DBL	V	ADD

Behaviour for ADD and for DBL V must be very close.

Software Implementation Result of Curve on 521 bits



Uniformization I

The costs for *ADD* and for *DBL* + *V* must be very similar.

Example: Weierstrass curve - projective

Verification:

$$V : Y^2Z = X^3 + aXZ^2 + bZ^3$$

$\Rightarrow$ Cost: $4M + 3S + 5A + 1 \times b$

ADD	DBL	DBL+V
$11M + 6S + 18A$	$5M + 6S + 14A$	$9M + 8S + 19A + 1 \times b$

Uniformization II

Example: Weierstrass curve - projective

- New verification:

$$Y^3Z = X^3Y + aXYZ^2 + bYZ^3$$

  $\Rightarrow$ Cost: $8M + 3S + 5A + 1 \times b$.
- Verification is included in *DBL*.
  $\Rightarrow$ Cost: $13M + 9S + 19A + 1 \times b$.
- Optimization using factorization.
  $\Rightarrow$ Cost: $11M + 6S + 18A + 1 \times b$.
- $1 \times b$ is added to *ADD*.

The costs for *ADD* and *DBL* + *V* are equal.

Summary

Good points

- This protection is equivalent to *double and add always* but at for smaller cost.
- Bit flip detection in all field elements.

Weakness

- Scalar is vulnerable to fault attacks.

Future work

- Uniformization for other coordinates.

Outline

Scalar Protection

Point verification does not protect the scalar from fault attack.

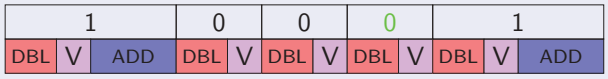
Example

1			0		0		0		1		
DBL	V	ADD	DBL	V	DBL	V	DBL	V	DBL	V	ADD

Scalar Protection

Point verification does not protect the scalar from fault attack.

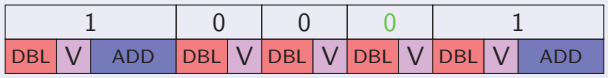
Example

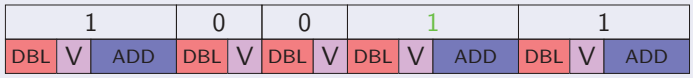


Scalar Protection

Point verification does not protect the scalar from fault attack.

Example

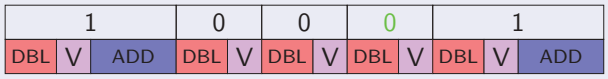


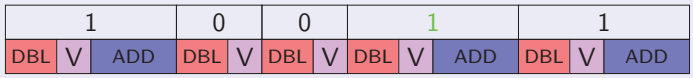


Scalar Protection

Point verification does not protect the scalar from fault attack.

Example





- Attack is not detected: current point is on curve but is wrong result.
- Proposed countermeasure: *Iteration Counter*.

Scalar Protection - Iteration Counter

Aim: Verify if the operation sequence is equivalent to key k.

Scalar Protection using Iteration Counter

Scalar Protection - Iteration Counter

Aim: Verify if the operation sequence is equivalent to key k.

First idea, naive idea:

- Count the number of *ADD* during computations.
  - Use a *cmp* register and a *ref* reference value.
  - If $k_i = 1$, add 1 to register.
- No detection in case 2 bit flips.

Scalar Protection using Iteration Counter

Scalar Protection - Iteration Counter

Aim: Verify if the operation sequence is equivalent to key k.

First idea, naive idea:

- Count the number of *ADD* during computations.
  - Use a *cmp* register and a *ref* reference value.
  - If $k_i = 1$, add 1 to register.
- No detection in case 2 bit flips.

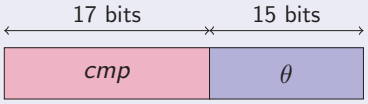
Second idea:

- Count the number of *ADD* combined with weight.
  - This weight is the iteration index.
  - If $k_i = 1$, add i to register.
- Problem: if $k_i = 0$, no electric activity.

Scalar Protection - Iteration Counter

Last idea:

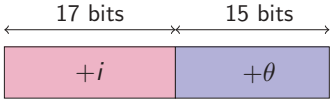
Divide the register in 2 parts.



- First part: counter *cmp*.
- Second part: noise θ.
- If $k_i = 1$, add iteration index i to *cmp*.
- If $k_i = 0$, add random number to θ.
- At the end: comparison between *cmp* and the reference counter *ref*.
- (17, 15) parameters has been selected for first implementation (work in progress).

Example - Double and Add

Require: P, $k = (k_{n-1}, \ldots, k_0)_2$. Data: *ref*.
Ensure: $k \cdot P$
  for $i = n - 1$ **to** 0 **do**
    $Q = 2 \cdot Q$
    $\theta =$ random number
    if $k_i = 1$ **then**
      $Q \leftarrow Q + P$
      $cmp \leftarrow cmp + i \cdot 2^{15}$
    else
      $cmp \leftarrow cmp + \theta$
    end if
  end for
  $cmp \leftarrow cmp >> 15$
  $cmp = ref$



Summary

Good points

- Cost: $\simeq n$ small integer add + small random number generator.
- Bit flip detection in the scalar.

Weakness

- Late detection.
- Stuck-at fault not detected $\Rightarrow$ safe error vulnerability.

Future works

- Key recoding.
- Key masking.

Outline

Evaluation I

What is implemented

- 3 curves:
  - Weierstrass, Edwards and Tripling-oriented Doche–Icart–Kohel (TDIK).
- 4 coordinates:
  - affine, projective, jacobian, standard.
- 4 algorithms:
  - double and add, Montgomery ladder, NAF, w-NAF ($w = 3, 4$).
- 3 point *verification* methods:
  - at the very end;
  - every ℓ iterations;
  - at each iteration ($\ell = 1$).

Evaluation II

Implementation target

- C language with GMP library.
- Intel core i7-5600U CPU @ 2.60GHz

Algorithms	Coordinates		
	Affine	Jacobian	Projective
Double and add	5.7%	20.1%	11.3%
Montgomery ladder	6.0%	20.3%	12.9%
NAF	6.1%	20.8%	12.7%
w-NAF	7.2%	24.5%	15.5%

Computation overheads in the worst case for Weierstrass curves.

Outline

Conclusion

What is done

- Bit flip attack detection.
  - on all field elements, on the scalar.
  - low cost.
  - adaptable for many curves.
- SPA resistant for projective coordinates.

Future works

- Stuck-at fault and safe error.
- Implementation on small processors (32, 16 bits).
- Uniformization for other curve.

Thank you for your attention.

Questions?

Centrality Indicators For Efficient And Scalable Logic Masking

Brice Colombier, Lilian Bossuet
Hubert Curien Laboratory, UMR CNRS 5516, University of Lyon
42000 Saint-Étienne - France
{b.colombier,lilian.bossuet}@univ-st-etienne.fr

David Hély
LCIS, Grenoble Institute of Technology
26000 Valence - France
david.hely@lcis.grenoble-inp.fr

***Abstract*—Modifying the logic at register transfer level can help to protect a circuit against counterfeiting or illegal copying. By adding extra gates, the outputs can be controllably corrupted. Then the circuit operates correctly only if the right value is applied to the extra gates. The main challenge is to select the best position for these gates, to alter the circuit's behaviour as much as possible. However, another major point is the computational efficiency of the selection process, which should be as good as possible for integration in EDA tools. State-of-the art methods, based on fault analysis, are very demanding and cannot cope with large netlists in a reasonable runtime. We propose to use centrality indicators instead. Centrality is used to identify the most significant vertices of a graph. We show that, when used to select the nodes to modify, they lead to low correlation between original and altered outputs while being computationally efficient. We give experimental results on combinational benchmarks and compare to other previously proposed heuristics. We show that this method is the only efficient selection heuristic which is able to handle large netlists and integrate smoothly into EDA tools.**

***Keywords*-centrality; logic masking;**

I. Introduction

Integrated circuits (IC) are increasingly complex, leading to outsourcing of manufacturing to overseas foundries and adoption of a design-and-reuse paradigm. Therefore, multiple actors take part in the realisation of an IC, from Intellectual Property (IP) core providers to system integrators and foundries. The designer must fully disclose the design for it to be further used, leading to the rise of counterfeiting [1].

Intellectual property protection means were proposed to thwart this threat [2] and some are based on RTL modifications [3]. We add our voice to authors of [4] to explicitly define the terms used for gate level modifications of netlists to achieve intellectual property protection. We use the terminology recently defined in [3], and focus on logic masking.

Logic masking aims at disrupting the outputs of the IC if the wrong masking key is applied. In order to disrupt the outputs as much as possible, some internal nodes of the netlist are modified to be controllably invertible. This is achieved by inserting XOR or XNOR gates on these nodes and connecting the other input of the added gates to the masking key bit inputs. Therefore, the IC operates correctly only if the correct masking key is applied. Otherwise, the added gates act as inverters and disrupt the circuit behaviour.

978-1-5090-6762-6/17/$31.00

Two research directions can essentially be observed in logic masking. The first trend aims at making the logic masking scheme resistant to various key-recovery attacks such as hill-climbing [5] or SAT [6]. It can be achieved by inserting an extra module before the key inputs of the masking scheme. Such module can either be an AES core [7] or a "hardware point function" which output is 1 only when the correct key is applied [8], [9]. The second research direction consist in finding the best location for the extra gates so that the outputs of the netlist are maximally corrupted when the wrong key is applied. Initially, their positions were randomly selected [10]. However, this led to very small drop in correlation, hence inefficient masking. More advanced heuristics were proposed later, based on fan-in/out [11], interference graphs [12], corruptibility [13] or fault-analysis [14]. Such methods improve the masking quality with an increasing computational effort. The latest heuristic to date [14] strongly disrupts the circuit outputs but becomes impractical to compute for netlist including more than a few thousand gates nodes. However, the masking scheme insertion method is meant to be integrated into the standard EDA design flow, where performance is crucial. Moreover, designs which are worth protecting are usually large. There is therefore a strong practical requirement for heuristics that offer a better trade-off between computational complexity and masking efficiency.

In this article, we propose to use centrality indicators from graph theory to select the nodes to mask. They leverage graph algorithms for efficiency and allow to reach low correlation between the normal and masked outputs, hence efficient masking. We start with a comparison of centrality indicators. We then compare with existing heuristics in two ways. First, by measuring the masking efficiency using correlation. Second, by comparing computation times required.

This article is organised as follows. Section II describes the use case, including the attacker model. Section III presents existing heuristics to select the nodes to modify for logic masking. Section IV discusses centrality indicators, and how they could be used in the considered context. Section V gives experimental results and compares with existing heuristics. Section VI discusses implementation issues. Section VII concludes the article. Our source code is fully available online[1]

[1]https://gitlab.univ-st-etienne.fr/b.colombier/centrality-based-logic-masking

for reproducibility.

II. Preliminaries

A. Use-case

Counterfeiting can be fought by protection means based on logic modification and meant to be integrated in EDA tools. Their intended users are fabless designers who wish to protect the intellectual property of their designs by modifying them prior to sending them to the foundry for manufacturing. Therefore, the proposed modification methods should have the following properties:

- **Efficiency:** the modifications should alter the outputs as much as possible when activated, leading to the lowest possible correlation between normal and modified outputs.
- **Complexity:** the modification process should be as computationally efficient as possible, in order to integrate smoothly in the design flow of EDA tools and be capable of handling large netlists which are worth being protected.

B. Attacker model

Since the aim of these protection method is to prevent counterfeiting, the attacker model we use is the following. An attacker owns two copies of the same circuit. One is fully functional, and seen as a black box. We then use the black box model for the circuit: the attacker can choose the inputs and observe the outputs. On the other hand, the attacker also owns a locked circuit, and wants to obtain the correct key for it. This occurs typically when a customer purchases circuits from regular and black market, and hopes to activate the ones obtained on the black market with the help of the legitimate circuits.

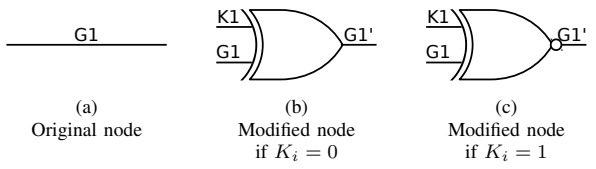
We assume the attacker cannot micro-probe the functional circuit to get the key. This requires a broader model, and is also much more costly from an attacker point of view.

III. State-of-the-art

A. Logic masking

As stated in [3], "*Logic masking consists in inserting XOR or XNOR gates in the data path of the logic circuit of a Boolean function in order to change the logic behaviour of the circuit if the wrong masking key is applied*".

First, the designer choses how many gates are to be modified in the netlist. To this end, an n-bit *masking word* is randomly generated. Next, a masking gate is inserted according to every bit of the masking word. If the bit is 0, then an XOR gate is inserted. If the bit is 1, then an XNOR gate is inserted. This is shown in Fig. 1.



(a) Original node (b) Modified node if $K_i = 0$ (c) Modified node if $K_i = 1$

Fig. 1. Modification of a node by logic masking.

If the correct masking word is applied the extra gates behave as buffers and the design operates normally. However, if a wrong masking word is applied, some of the extra gates will behave as inverters, effectively disrupting the circuit behaviour and corrupting the outputs.

B. Nodes selection heuristics

The following nodes selection heuristics select n nodes in the netlist, on which additional masking gates are to be inserted.

1) Random: The most basic way to select the nodes is random selection. This was the first proposed method, in EPIC [10]. This is fast, since no computation is required.

2) Fan-in/Fan-out cones: In 2009, authors of [11] proposed the first heuristic which improves the selection. It is based on the number of netlist nodes that are in the fan-in and fan-out cones of every other node in the netlist. The exact metric is given in Equation (1): FI and FO are the number of nodes in the fan-in and fan-out cones for every node. FI_{max} and FO_{max} are the maximum values of FI and FO observed in the netlist. w_1 and w_2 are normalisation weights which are set to 0.5. The nodes that maximise this metric are modified.

$$M_{node} = \left(\frac{w_1.FO}{FO_{max}} + \frac{w_2.FI}{FI_{max}}\right) \times \frac{FO.FI}{FI_{max}.FO_{max}} \quad (1)$$

According to this metric, the nodes with the greatest number of nodes in their fan-in/out cones are the most significant.

3) Interference graph: In [12], the random method has been improved. Initially, 10% of the masking gates are inserted randomly, to initiate the procedure. An interference graph is then built from the relative positions of the gates. The interference graph represents how the inserted gates interact with one another. For example, two masking gates placed in a row or two gates that converge to the same node are represented differently in the interference graph. Then, for every node of the netlist a metric is computed with respect to the existing masking gates, from the interference graph. The node that maximises this metric is selected, added to the interference graph and a masking gate is inserted on it. The process is then repeated again until all the masking gates are inserted.

4) Corruptibility: The authors of [12] improved their interference graph-based heuristic in [13] by adding a so-called *corruptibility* metric. This ensures that non-resolvable gates which are selected after analysing the interference graph corrupt the outputs as much as possible. Corruptibility is computed as the ratio of output patterns that differ between the normal and masked behaviour of the circuit. A node then has a high corruptibility if modifying it for logic masking changes the outputs most of the time. Computing the corruptibility requires to simulate the netlist using a dedicated tool. In [13], the authors used a fault-simulation tool, and computed corruptibility by observing one thousand input/output patterns. Such tools are usually computationally heavy.

5) Fault analysis: This is the latest proposed heuristic to date [14]. Based on fault simulation, it acts by computing the *Fault Impact* for every node of the netlist, given in Equation (2). NoP_0 is the number of patterns that can detect that the node is

stuck-at-0. NoO_0 is the number of output bits affected by this stuck-at-0 fault. NoP_1 and NoO_1 are similar but for stuck-at-1 faults.

$$\text{Fault Impact} = NoP_0.NoO_0 + NoP_1.NoO_1 \quad (2)$$

By considering both stuck-at-0 and stuck-at-1 faults, authors select the node with the greatest impact on the outputs. However, this selection heuristic is based on fault simulation, hence it remains computationally heavy. Moreover, it is recomputed every time a gate is added.

C. Netlist to graph conversion

We chose to convert the netlists to directed acyclic graphs using the same method as in [3]. Vertices are the netlist nodes and edges are logic functions connecting the nodes. A toy example is shown in Fig. 2.

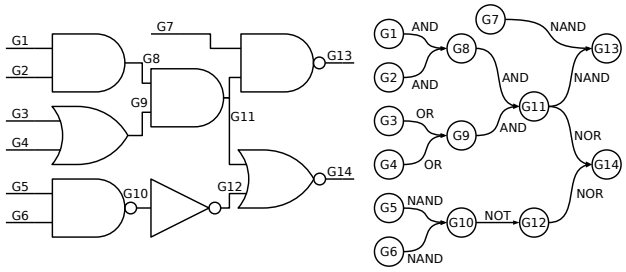


Fig. 2. A netlist and the equivalent directed acyclic graph. Netlist nodes are converted to vertices and logic functions to edges.

IV. Centrality indicators

Centrality indicators determine which vertices are the most significant in a graph. This "significant" term is very broad, and different centrality indicators perform better at identifying the "significant" vertices in different contexts. From a logic masking point of view, significant nodes are the ones for which a modification alters the most the circuit operation. Intuitively, those nodes are the ones through which a lot of information transits, from the inputs to the outputs of the circuit. Among centrality indicators, there are *local* and *global* ones. Local centrality indicators are computed according to the vertices found in the direct neighbourhood of the considered vertex. On the other hand, global indicators take the whole graph into consideration and thus they are more suited in our use case. We start by examining common global centrality indicators, before considering more sophisticated ones that are well suited to identify significant nodes in terms of "information transit".

Normalised centrality indicators: In the literature, some indicators are normalised according to the number of vertices in the network. Depending on how vertices are considered, the final indicator value can be divided by the total number of vertices. Since we are interested in the relative value of centrality for the vertices, this normalisation is not necessary. We then use only the non-normalised versions of the following indicators.

A. Closeness centrality

Closeness centrality is defined as the inverse of farness [15]. For a given vertex v, the farness is the sum of the distances from v to the other graph vertices (see Equation (3), where $d(v, y)$ stands for the distance between vertices v and y).

$$C_C(v) = \frac{1}{\sum\limits_{y:y\in V} d(v, y)} \quad (3)$$

Therefore, a vertex is significant in the sense of closeness centrality if it is the closest to all the other vertices in the graph. Practically, the vertices that have the highest closeness centrality are "in the middle" of the netlist.

It is more interesting in the case of logic masking than degree centrality because it is a global centrality indicator. Therefore, it is influenced by the graph structure.

B. Betweenness centrality

Betweenness centrality [16] of node v is given by the ratio of shortest paths between all other vertices in the graph that traverse v. This is given in Equation (4), in which σ_{st} stands for the number of shortest paths that go from s to t, and σ_{svt} stands for the total number of shortest paths that go from s to t through v.

$$C_B(v) = \sum_{s\neq t\neq v:\{s,t,v\}\in V} \frac{\sigma_{svt}}{\sigma_{st}} \quad (4)$$

Intuitively, for a netlist, betweenness centrality will be the highest for nodes that are on the shortest paths from inputs to outputs. This is interesting for logic masking, since those nodes are typically the ones for which masking will have the greatest impact on information transiting from inputs to outputs. The main drawback of this indicator, however, is that it only accounts for shortest paths. As pointed out in [17], this is quite a restrictive constraint. Indeed, it assumes that information only flows along shortest paths, which is certainly not always true.

Alternative centrality indicators based on current flow have been proposed. By assuming that information behaves in the same way as electrical current, the authors of [17], [18] account for the fact that it can split and spread in the network. This is discussed in the next subsections.

C. Current-flow betweenness centrality

Current-flow betweenness centrality [18] considers the graph as an electrical network. Vertices are converted to nodes, and the edges connecting them are replaced by unit resistors. Pairs of vertices are successively picked as current input and output. The sum of current that flows through node v for all the pairs of vertices picked gives the current-flow betweenness centrality for this node. This is shown in Equation (5), where $I_v^{(st)}$ is the current flowing through node v when s is the input and t is the output.

$$C_{CFB}(v) = \sum_{s\neq t:\{s,t\}\in V} I_v^{(st)} \quad (5)$$

This measure of centrality is more subtle than betweenness centrality. Indeed, instead of considering only shortest paths

between vertices, the current is inversely proportional to the path length. This is a more precise assumption about the way information spreads in a network.

1) Approximate current-flow betweenness centrality: For current-flow betweenness centrality, running time and space requirements rapidly become prohibitive for large graphs. An approximate version has been proposed in [17]. Instead of using all the s and t pairs in the graph as current inputs and outputs, they show that using a smaller number of randomly selected pairs leads to a good approximation. This is an interesting point in the considered use-case. Large netlists can then be analysed, by relaxing precision.

D. Current-flow closeness centrality/Information centrality

Current-flow closeness centrality was proposed in [17], and is equivalent to information centrality [19]. Instead of using distance between nodes as a measure of closeness, it proceeds similarly to current-flow betweenness centrality. First, the graph is converted to an equivalent electrical network with edges replaced by unit resistors. Afterwards, farness is the difference of potential (voltage) between the two nodes. It is the equivalent resistance between the two nodes (see Equation (6)). Thus it also accounts for paths which are not the shortest ones. All the paths between two nodes are considered, contributing to the overall equivalent resistance depending on their length.

$$C_{CFC}(v) = \frac{1}{\sum\limits_{y:y\in V} p(v) - p(y)} = \frac{1}{\sum\limits_{y:y\in V} R_{eq}(v,y)} \quad (6)$$

V. Experimental results

The centrality indicators were computed using Python *igraph* [20] and *NetworkX* [21] libraries. Our workstation embeds an Intel Core i5-4570 operating at 3.20GHz and 16GB of RAM. Experimental results were obtained with ITC'99 [22] and EPFL [23] combinational benchmarks. We restrict the size of the benchmarks from 1k to 100k gates. Note that this is the only method demonstrated on large benchmarks, up to 100k gates, when [10], [11], [14] do not exceed a few thousand gates. For large benchmarks, come centrality indicator computations ran out of memory and are not presented. Moreover, we fix a timeout limit for computation of 1h.

A. Masking efficiency evaluation

The Hamming distance criterion used in previous articles to evaluate the masking efficiency is not suited as detailed in [3]. As stated in [13], "*We need maximum corruption, and thus minimum correlation at the outputs*". Therefore, the masking efficiency E_m is evaluated by computing the quadratic mean of the Pearson's correlation coefficient (see Eq. (7)) obtained between the normal and masked mode for every output bit (see Eq. (8)).

$$r(x,y) = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}} \quad (7)$$

$$E_m = \sqrt{\frac{1}{n}\sum_{o\in outputs} r^2(o_{normal}, o_{masked})} \quad (8)$$

The output values are obtained by applying 10k random vectors at the key and primary inputs of the netlist. For each benchmark and area overhead, the most efficient centrality indicator is in bold face.

TABLE I
E_m VALUES FOR SELECTION HEURISTICS BASED ON CENTRALITY INDICATORS AND DIFFERENT LOGIC RESOURCES OVERHEADS

Benchmark	#gates	Centrality indicator	Logic resources overhead 1%	5%	10%
adder	∼1k	B	0.97	0.86	0.70
		C	0.98	0.94	0.91
		C-FB	0.94	**0.73**	**0.58**
		C-FC	0.96	0.95	0.93
		AC-FB	**0.91**	0.75	0.68
i2c controller	∼1k	B	0.97	0.91	0.85
		C	0.98	0.93	0.90
		C-FB	0.28	**0.19**	**0.17**
		C-FC	**0.27**	0.20	0.37
		AC-FB	0.29	0.28	0.23
sine	∼5k	B	0.23	0.26	0.22
		C	0.26	0.20	0.23
		C-FC	0.21	**0.01**	**0.01**
		AC-FB	**0.10**	0.11	**0.01**
b14_1_C	∼5k	B	0.92	0.61	0.50
		C	0.74	0.65	0.48
		AC-FB	**0.73**	**0.36**	**0.34**
b15_1_C	∼10k	B	0.82	0.57	**0.45**
		C	0.81	**0.61**	0.48
		AC-FB	**0.77**	0.64	0.85
round-robin arbiter	∼10k	B	0.96	**0.84**	**0.69**
		C	0.94	0.86	0.83
		AC-FB	**0.94**	0.88	0.83
memory controller	∼50k	B	0.98	0.94	0.88
		C	0.98	**0.91**	**0.83**
divisor	∼50k	B	0.65	0.64	0.64
		C	0.65	0.64	0.64
b18_1_C	∼100k	B	0.95	0.80	**0.63**
		C	0.95	0.80	0.65

B: betweenness
C: closeness
C-FB: current-flow betweenness
C-FC: current-flow closeness
AC-FB: approximated current-flow betweenness

The masking efficiency values E_m obtained are shown in Table I, in which three logic resources overheads are considered, 1%, 5% and 10%. The overhead is computed as the percentage of extra gates added to the design. E_m values differ greatly depending on the benchmark. For some of them, the correlation drops very fast, even at low overhead. This occurs for benchmarks in which outputs are strongly correlated, such as *sine*. On the other hand, some benchmarks make it very hard to reduce the correlation coefficient, even with a 10% overhead.

The centrality indicators differ in effectiveness. However, the ones based on current-flow are the most efficient is the majority of cases. For the largest benchmark, b18_1_C, which comprises 100k gates, the correlation drops to 0.63 for 10% area overhead. This shows that the masking is efficient, even

on very large netlists.

Increasing the overhead obviously reduces correlation since inserting more masking gates increases the masking efficiency.

Additionally, we estimated by simulation the corruptibility of the outputs when centrality indicators are used to select the nodes to modify. For all the circuits and all centrality indicators, when an incorrect key is applied, the normal and masked outputs were systematically different.

B. Computation time

Fig. 3 shows on a log-log scale how computation time varies with respect to the number of nodes in the netlist. The dark grey line is the baseline for computation time. It is the time required to only build the graph as described in Subsection III-C.

Centrality indicators are efficient to compute in general, although the centrality indicators based on current-flow require more time. However, even for a very large benchmark of 100k gates, computing betweenness and closeness centrality is possible in less than an hour on our desktop workstation. Surely this could be improved with a dedicated server. Moreover, recent research [24] shows that centrality indicators can be computed faster in a distributed manner. Computation time does not depend on the chosen overhead, since the chosen centrality indicator must be computed for all the nodes of the graph in all the cases.

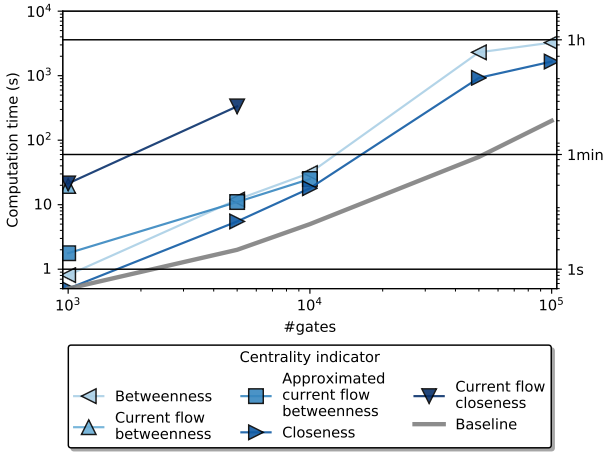


Fig. 3. Computation time required using different heuristics for selection with a 5% logic resources overhead. The baseline is random selection.

C. Comparison with existing heuristics

Fig. 4 illustrates the trade-off between correlation reduction and computation time. The baseline for computation time is random selection as it is the simplest method, thus the fastest to compute. The most efficient heuristics are closer to the origin, since they are the fastest to compute and the most efficient at reducing correlation.

Other heuristics can be broadly classified into two categories. First, fault-analysis based selection [14] can reduce correlation significantly, down to 0.2. However, this selection

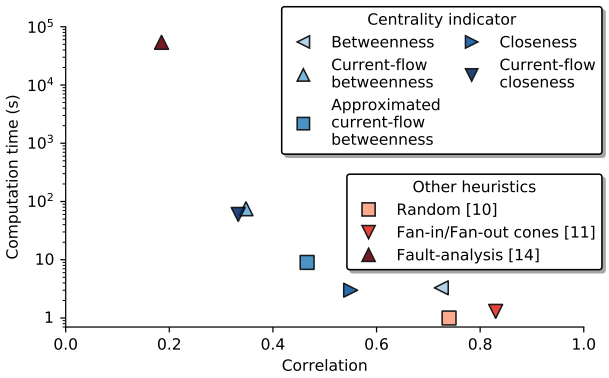


Fig. 4. Trade-off between computation time and correlation reduction. The logic resources overhead is 5-6%. The correlation and computation time values are obtained after averaging over all benchmarks, except for [14] for which only three small benchmarks from the original article are considered.

heuristic is very computationally expensive. Authors report that "*This method took two hours to encrypt the C7552 circuit*". This circuit only has 3,500 nodes. Therefore, fault-analysis based selection is highly impractical and cannot cope with large netlists, which are typically the ones worth protecting against counterfeiting. Emulation has been proposed in [25] to speed-up the process but it requires a very large FPGA for implementation since it increases the size of the original a lot. Moreover, the correlation value of 0.2 is obtained from only three combinational benchmarks, which are relatively small. Nothing shows that this low correlation would be observed on larger benchmarks such as the ones we used. On the other hand, random [10] and fan-in/fan-out cones [11] methods are rapidly computed. However, as visible in Fig. 4, the correlation remains very high. Therefore, they do not achieve efficient masking.

Overall, existing heuristics are either efficient at reducing correlation, but complex to compute, or easy to compute but inefficient at reducing correlation. In contrast, centrality indicators can reduce correlation down to 0.4 on average. Moreover, they are much more computationally efficient than fault-analysis based selection, since they run 1,000 times faster on average. Among centrality indicators, the most efficient are the ones based on current flow. They are the closest to the origin, and reduce correlation efficiently while being computationally practical. Therefore, centrality indicators offer a better trade-off than state-of-the-art heuristics between masking efficiency and computational complexity.

VI. Discussion

A. Impact on maximum operating frequency

When choosing the nodes to modify, critical paths can be excluded. This way, the impact of logic masking on the operating frequency is minimised. Masking efficiency would not be affected much since critical paths are marginal.

B. Sequential circuits

When masking sequential circuits, combinational parts must be isolated. Flip-flop inputs are converted to graph output nodes, and the flip-flop outputs to graph input nodes [4].

C. Scalability

The results we provide here for computation time are obtained on a standard desktop workstation. In order to improve performances, a dedicated server with more memory could be used to provide more computing power and analyse larger netlists. Another option to improve scalability is to compute the centrality indicator in parallel. Recent research [24] highlight the fact that current flow-based centrality indicators, which are usually the most efficient for logic masking, could be computed faster. Other heuristics based on interference graph [12] or fault analysis [14] are intrinsically sequential since they require the masking metric to be recomputed every time a node is modified.

D. Controllability and distance to inputs and outputs

For most of the modified benchmarks, inspection shows that the inserted gates are as close to the inputs as they are to the outputs. They are then approximately in the middle of the netlist. This is a good point against reverse-engineering. Indeed, if the extra gates are embedded deeper in the netlist, they are harder to uniquely identify and disable.

The distance to inputs and outputs is closely related to the controllability of the nodes. In order to make sure that the modified nodes are hard to control, one can set a threshold on their controllability. The controllability value can be computed very fast. By ensuring that the controllability of the selected nodes is high enough, the key value is much harder to reveal on the outputs of the circuit by sensitisation attack [13].

VII. Conclusion

We proposed to use centrality indicators to select the nodes modified by logic masking. On the one hand, it reduces correlation effectively and is faster to compute than state-of-the-art effective heuristics. On the other hand, compared to other computationally-efficient heuristics, it reduces correlation significantly more. Overall, it provides a better trade-off between masking efficiency and computational complexity, and is the only realistic candidate for integration in EDA tools dealing with large and complex netlists.

Acknowledgements

The work presented in this paper was realised in the frame of the SALWARE project number ANR-13-JS03-0003 supported by the French "Agence Nationale de la Recherche" and by the French "Fondation de Recherche pour l'Aéronautique et l'Espace", funding for this project was also provided by a grant from "La Région Rhône-Alpes".

References

[1] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit integrated circuits: A rising threat in the global semiconductor supply chain," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1207–1228, 2014.

[2] B. Colombier and L. Bossuet, "Survey of hardware protection of design data for integrated circuits and intellectual properties," *IET Computers & Digital Techniques*, vol. 8, no. 6, pp. 274–287, Nov. 2014.

[3] B. Colombier, L. Bossuet, and D. Hély, "From secured logic to IP protection," *Elsevier Microprocessors and Microsystems*, vol. 47, pp. 44–54, 2016.

[4] S. M. Plaza and I. L. Markov, "Protecting integrated circuits from piracy with test-aware logic locking," in *International Conference on Computer Aided Design*, San Jose, CA, USA, Nov. 2014.

[5] ——, "Solving the third-shift problem in IC piracy with test-aware logic locking," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 961–971, 2015.

[6] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *IEEE International Symposium on Hardware Oriented Security and Trust*, Washington, DC, USA, May 2015, pp. 137–143.

[7] M. Yasin, J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1411–1424, 2015.

[8] M. Yasin, B. Mazumdar, J. J. V. Rajendran, and O. Sinanoglu, "Sarlock: SAT attack resistant logic locking," in *IEEE International Symposium on Hardware Oriented Security and Trust*, McLean, VA, USA, May 2016, pp. 236–241.

[9] Y. Xie and A. Srivastava, "Mitigating SAT attack on logic locking," in *International Conference on Cryptographic Hardware and Embedded Systems*, Santa Barbara, CA, USA, Aug. 2016, pp. 127–146.

[10] J. A. Roy, F. Koushanfar, and I. Markov, "Ending piracy of integrated circuits," *Computer*, vol. 43, no. 10, pp. 30–38, 2010.

[11] R. S. Chakraborty and S. Bhunia, "HARPOON: an obfuscation-based soc design methodology for hardware protection," *IEEE Transations on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 10, pp. 1493–1502, 2009.

[12] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Annual Design Automation Conference*, San Francisco CA, USA, Jun. 2012, pp. 83–89.

[13] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *ACM Conference on Computer & communications security*, Berlin, Germany, Nov. 2013, pp. 709–720.

[14] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 410–424, 2015.

[15] G. Sabidussi, "The centrality index of a graph," *Psychometrika*, vol. 31, no. 4, pp. 581–603, 1966.

[16] J. M. Anthonisse, "The rush in a directed graph," *Mathematische Besliskunde*, no. BN 9/71, pp. 1–10, 1971.

[17] U. Brandes and D. Fleischer, "Centrality measures based on current flow," in *Annual Symposium on Theoretical Aspects of Computer Science*, vol. 3404, Stuttgart, Germany, Feb. 2005, pp. 533–544.

[18] M. E. J. Newman, "A measure of betweenness centrality based on random walks," *Social Networks*, vol. 27, no. 1, pp. 39–54, 2005.

[19] K. Stephenson and M. Zelen, "Rethinking centrality: Methods and examples," *Social Networks*, vol. 11, no. 1, pp. 1–37, 1989.

[20] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal Complex Systems*, vol. 1695, no. 5, pp. 1–9, 2006.

[21] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Python in Science Conference*, Pasadena, CA USA, Aug. 2008, pp. 11–15.

[22] S. Davidson, "ITC'99 benchmark circuits - preliminary results," in *IEEE International Test Conference*, Atlantic City, NJ, USA, Sep. 1999, p. 1125.

[23] L. Amarú, P.-E. Gaillardon, and G. D. Micheli, "The EPFL combinational benchmark suite," in *International Workshop on Logic & Synthesis*, Mountain View, CA, USA, Jun. 2015.

[24] A. Lulli, L. Ricci, E. Carlini, and P. Dazzi, "Distributed current flow betweenness centrality," in *International Conference on Self-Adaptive and Self-Organizing Systems*, Cambridge, MA, USA, Sep. 2015, pp. 71–80.

[25] S. Gören, C. C. Gürsoy, and A. Yildiz, "Speeding up logic locking via fault emulation and dynamic multiple fault injection," *Journal of Electronic Testing*, vol. 31, no. 5-6, pp. 525–536, 2015.

Low-complexity DPA Countermeasure for Resource-Constrained Embedded McEliece Implementation

Martin Petrvalský
Miloš Drutarovský
Dept. of Electronics
& Multimedia Communications,
Technical University of Kosice,
Park Komenskeho 13,
041 20 Kosice, Slovakia
Email: {martin.petrvalsky,
milos.drutarovsky}@tuke.sk

Tania Richmond
Institut de Mathématiques
B.P. 20132,
83957, La Garde, France.
Email: tania.richmond@univ-tln.fr

Pierre-Louis Cayrel
and Viktor Fischer
Laboratoire Hubert Curien,
Rue du Prof. Benoît Lauras, 18,
42000, Saint-Étienne, France.
Email:{pierre.louis.cayrel,
fischer}@univ-st-etienne.fr

***Abstract*—In this paper, we present an efficient countermeasure against side-channel attacks on the McEliece public-key cryptosystem. Firstly, we deploy a novel correlation based differential power analysis attack combined with a chosen ciphertext attack targeting a secure implementation of the McEliece cryptosystem. We demonstrate that a part of a private key (permutation matrix) can be recovered using the attacks. Furthermore, we show that a revelation of the permutation matrix possess a critical threat to the cryptosystem. The cryptosystem and its variations are implemented on a 32-bit ARM-based microcontroller. We provide details and results of the attack using power consumption measurements of the hardware. As a main contribution, we provide an experimental results of a novel efficient countermeasure against the attack. The countermeasure (which is a derivate of a masking technique) uses properties of the McEliece cryptosystem in order to reduce the complexity of the masking technique. Our new method does not require a large amount of random bits and it reduces computational time compared to the regular masking technique. These properties can be profitable for low-cost constrained and embedded devices.**

***Keywords*—Correlation based differential power analysis, masking countermeasure, McEliece public-key cryptosystem, secure bit permutation, side-channel attack.**

I. Introduction

Requirements for low-cost embedded device security rises its importance in the context of the Internet of Things (IoT) [1]. Cryptographic algorithms are implemented on these devices in order to provide needed security features. If we are thinking in a long-term manner, currently used public-key cryptosystems (PKCs) such as RSA [2] and ECC [3] will be vulnerable with advances in quantum computing. The solution for this problem are PKCs which will be theoretically secure even in the post-quantum era (post-quantum PKCs [4]). McEliece PKC [5] is one of the promising cryptosystems which is invariant against quantum computing algorithms. It is based on error-correcting codes and it was proposed by McEliece in 1978. The cryptosystem offers interesting properties - fast computation, favorable theoretical complexity which make the cryptosystem suitable for currently used low-cost embedded devices such as microcontrollers.

Theoretical security of the algorithm, especially for embedded devices, is often not sufficient in order to protect the device against possible attacks. Side-channel attacks (SCAs) exploit a physical phenomenon of an implementation, e.g. running time in software or power consumption in hardware. The first SCA against the McEliece PKC was proposed in 2008 [6] and more papers have followed during the last years. Most of those attacks target the Patterson's algorithm [7] and they are based on timing attack [6], [8]–[11]. We are focused on power consumption attacks on the McEliece PKC of which simple power analyses (SPAs) have been proposed in [12], [13] and differential power analyses (DPAs) attacks have been proposed in [14]–[16].

Regarding [12, Section 3], there are four profiles used for the first steps of the McEliece decryption. Profiles III and IV do not require permutation algorithm and thus they are more secure. However, the first step in profiles I and II permutes input ciphertexts which is convenient for some applications, e.g. in [6] or for low-cost embedded and constrained devices. In this work, we are focused on the profiles I and II where the permutation algorithm must be implemented. We focus on a development of a countermeasure which efficiently complicates (not completely avoids) side-channel attacks with emphasis on a low computational and memory requirements.

Compared to our previous work [15], [16], we present a novel correlation based DPA (CBDPA) attack on a secure implementation of a bit permutation in McEliece PKC. Furthermore, we provide new details on a possible security threat on revealed permutation matrix. In addition, we design a new countermeasure against side-channel attacks and we provide experimental results of the secure implementation of the bit permutation and its security against DPA attacks.

The paper is organized as follows. We start with a short review on Goppa codes and the McEliece PKC in Section II. In Section III, we present a CBDPA attack on the protected

McEliece ciphertext permutation and a syndrome computation given in [15]. Furthermore, in Section IV, we propose a new low-complexity countermeasure against CBDPA attacks. We show experimental results of our work in Section V. Finally, we conclude the paper in Section VI.

II. Theoretical Background

A. Goppa Codes

Goppa proposed a large class of linear error-correcting codes in 1970 [17], [18]. However our interest is focused exclusively on (irreducible) binary Goppa codes, that are available for encryption. For the sake of simplicity, we will call them Goppa codes.

Definition 1: Let m and t be positive integers. Let a support $\mathcal{L} = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ represent a subset of $\mathbb{F}_{2^m}$ such that the α_i are pairwise distinct elements, so $n \leqslant 2^m$. Given a monic (irreducible) polynomial $g(x) \in \mathbb{F}_{2^m}[x]$ such that $\deg(g) = t$ and $g(\alpha_i) \neq 0$, the (irreducible) Goppa code $\Gamma(\mathcal{L}, g)$ is defined as: $\Gamma(\mathcal{L}, g) =$
$\{C = (C_1, C_2, \ldots, C_n) \in \mathbb{F}_2^n \mid \sum_{i=1}^{n} \frac{C_i}{x \oplus \alpha_i} \equiv 0 \mod g(x)\}$.

$$\mathcal{S}_C(x) = \sum_{i=1}^{n} \frac{C_i}{x \oplus \alpha_i}. \quad (1)$$

We call the syndrome polynomial, a polynomial associated to $C \in \mathbb{F}_2^n$ given by (1). To decode a binary Goppa codeword containing errors, one commonly adopted solution is to use the so-called Patterson's algorithm [7]. We will focus on the first step of this algorithm consisting of computing a product between the codeword C with at most t errors and a parity-check matrix of the Goppa code denoted $\mathcal{H}$, i.e. $S = C \cdot \mathcal{H}^T$. The result of this operation is called the syndrome and it can be viewed as a polynomial as $S_C(x) = [x^{t-1}, \ldots, x, 1] \cdot S$, i.e. (1).

B. The McEliece Cryptosystem

McEliece proposed the first public-key code-based encryption scheme in 1978 [5]. The McEliece PKC, using Goppa codes as in the original paper, is performed using the three following algorithms - key generation, plaintext encryption and ciphertext decryption.

Key Generation. The key generation consists of the determination of the Goppa code according to Definition 1 given in Section II-A. Since Goppa codes are linear, they can be generated by a so-called $k \times n$ generator matrix denoted $\mathcal{G}$ (usual values of n are 1024, 2048, ...). We randomly choose a non-singular $k \times k$ matrix $\mathcal{S}$ and a $n \times n$ permutation matrix $\mathcal{P}$. Then we compute the public $k \times n$ generator matrix given by $\tilde{\mathcal{G}} = \mathcal{S} \cdot \mathcal{G} \cdot \mathcal{P}$. The key generation provides the private key $\mathsf{sk} = (\Gamma(\mathcal{L}, g), \mathcal{S}, \mathcal{P})$ and the public key $\mathsf{pk} = (m, t, \tilde{\mathcal{G}})$. Notice that k and n are also public because $\tilde{\mathcal{G}}$ is a $k \times n$ matrix.

Plaintext Encryption. During the plaintext encryption, the message M is encrypted using the public generator matrix. This operation can be expressed by $C = M \cdot \tilde{\mathcal{G}}$. Then an error vector E of length n and weight t is randomly selected and added to the codeword C, giving the ciphertext $\tilde{C} = C \oplus E$.

Ciphertext Decryption. At first, the product $\tilde{C}_p = \tilde{C} \cdot \mathcal{P}^{-1}$ is computed during the decryption of the ciphertext $\tilde{C}$. The attack described in Section III targets this phase of the ciphertext decryption. This step is leading to a codeword containing an error, i.e. : $\underbrace{M \cdot \mathcal{S} \cdot \mathcal{G}}_{\text{codeword}} \oplus \underbrace{E \cdot \mathcal{P}^{-1}}_{\substack{\text{error vector} \\ \text{of weight } t}}$. Next, a decoding algorithm (the Patterson's algorithm in our case) must be applied on the obtained secret code. Let $\mathcal{G}_r^{-1}$ be the $\mathcal{G}$ right-side inverse, i.e. $\mathcal{G} \cdot \mathcal{G}_r^{-1} = \mathcal{I}_k$, where $\mathcal{I}_k$ is the $k \times k$ identity matrix. Thereafter the obtained codeword $M \cdot \mathcal{S} \cdot \mathcal{G}$ is multiplied by $\mathcal{G}_r^{-1}$ on the right-side, in order to find $\tilde{M} = M \cdot \mathcal{S}$. Finally, we compute $M = \tilde{M} \cdot \mathcal{S}^{-1}$ to recover the plaintext.

III. CBDPA of the Private Permutation

We attack a secure implementation of a bit permutation algorithm from [15] (recalled in this paper in Algorithm 1). In [16], different secure implementation of the bit permutation [6] is successfully attacked. Side-channel vulnerabilities can be found in both implementations. They leak an information about the private permutation matrix $\mathcal{P}^{-1}$ in the McEliece PKC decryption.

In Algorithm 1, the vulnerable operation is the state 9. If a bit of the permuted ciphertext $\tilde{C}_{p_i} = 0$ then $tmp = 0$; if $\tilde{C}_{p_i} = 1$ then $tmp = (FFF \ldots F)_{hex}$. Using multiple measurements with different inputs (e.g. 500 measurements) we are able to detect the exact time when particular bits are handled in tmp variable. Time of the particular bit handling corresponds to the current location in the vector $\tilde{C}_p$. By comparing input vector $\tilde{C}$ and permuted vector $\tilde{C}_p$ extracted from the CBDPA attack, it is possible to extract the whole permutation matrix $\mathcal{P}^{-1}$. We attack an implementation [15] using $n = 1024$. Compared to the the attack in [16], we extract a permutation matrix $\mathcal{P}$ by attacking a secure syndrome computation (parity-check matrix multiplication) which is scheduled after the bit permutation.

Algorithm 1 Bit permutation and a secure syndrome computation (multiplication by the parity-check matrix) $S = (\tilde{C} \cdot \mathcal{P}^{-1}) \cdot \mathcal{H}^T$ (from [15])

Require: Private permutation matrix $\mathcal{P}^{-1}$ represented by lookup-table $t^{\mathcal{P}}$, private parity-check matrix $\mathcal{H}^{\mathcal{T}}$, ciphertext vector $\tilde{C}$ and mask M.
Ensure: Syndrome S
1: $\tilde{C}_p = 0$
2: **for** $i = 0$ **to** $n - 1$ **do**
3: **if** $\tilde{C}_i = 1$ **then**
4: $\tilde{C}_p = \tilde{C}_p \oplus (1 \ll t_i^{\mathcal{P}})$
5: **end if**
6: **end for**
7: $S = M$
8: **for** $i = 0$ **to** $n - 1$ **do**
9: $tmp = unsigned(0 - \tilde{C}_{p_i})$
10: $S = S \oplus (tmp \,\&\, H_i^T)$
11: **end for**
12: $S = S \oplus M$
13: **return** S

A. Methods of the CBDPA Attack

We deploy the CBDPA attack based on the Pearson's correlation coefficient [19] with known input ciphertext. The workflow of the CBDPA is depicted in Fig. 1. We apply a Hamming weight of individual bits leakage model ($H_i \in \{0, 1\}$). We choose the Hamming weight model since it accurately

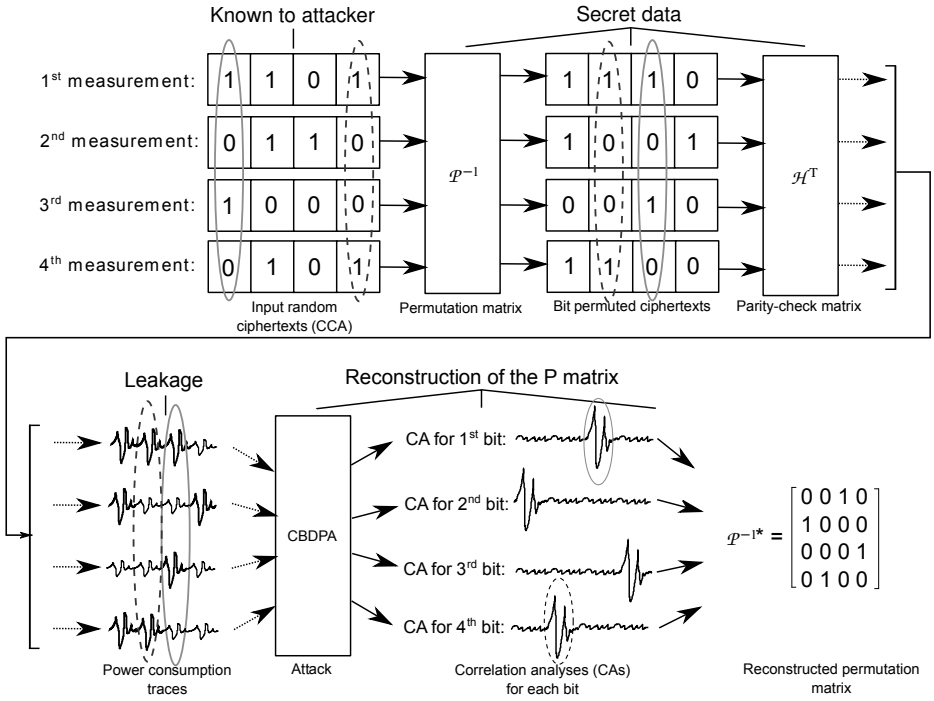


Figure 1. Main steps of the CBDPA attack on the secure bit permutation. We input random ciphertexts while the permutation matrix $\mathcal{P}^{-1}$ remains the same. We measure a leakage on a syndrome computation operation which is scheduled after the bit permutation (the attack on [15]).

represents the leakage of the attacked device. We use (2) for correlation analyses:

$$r_{H,X}(\eta) = \frac{\sum_{i=1}^{N}[(X_i(\eta) - \bar{X}(\eta))(H_i - \bar{H})]}{\sqrt{\sum_{i=1}^{N}[X_i(\eta) - \bar{X}(\eta)]^2 \sum_{i=1}^{N}(H_i - \bar{H})^2}} \quad (2)$$

where $r_{H,X}(\eta)$ is the Pearson's correlation coefficient for η-th sample (measured during execution of the cryptographic algorithm), N is a number of measured traces, $X_i(\eta)$ is a value of η-th sample measured during i-th measurement (i-th trace), $\bar{X}(\eta)$ is a mean value of corresponding η-th samples (from all traces), H_i is a hypothesis of power consumption for one bit of input data corresponding with i-th measurement (i-th trace) and $\bar{H}$ is a mean value of all hypotheses H_i.

B. A threat of the revealed $\mathcal{P}$

Assuming one knows the private permutation matrix $\mathcal{P}$ and the support $\mathcal{L}$ is in lexicographic order (or any other one, but known by the attacker). That means the only remaining part of the private key still unknown is the Goppa polynomial. Indeed, the knowledge of the scrambling matrix $\mathcal{S}$ is useless because $\mathcal{G}$ and $\mathcal{S} \cdot \mathcal{G}$ generate the same code. Multiply a generator matrix by an invertible matrix on the left side is the only way to get another generator matrix of the same code.

Notice that a parity-check matrix $\mathcal{H}$ of the Goppa code can be computed as a product between the t-rows Vandermonde matrix $\mathcal{Y}$ of support $\mathcal{L}$ elements and the diagonal matrix $\mathcal{Z}$ with the inverse polynomial evaluation of g on all elements in the support $\mathcal{L}$, where each component is an m-bit vector over $\mathbb{F}_2$. That is, over $\mathbb{F}_{2^m}$, $\mathcal{H}$ looks like:

$$\mathcal{H} = \underbrace{\begin{pmatrix} 1 & 1 & \dots & 1 \\ \alpha_1 & \alpha_2 & \dots & \alpha_n \\ \vdots & \vdots & \dots & \vdots \\ \alpha_1^{t-1} & \alpha_2^{t-1} & \dots & \alpha_n^{t-1} \end{pmatrix}}_{\mathcal{Y}} \cdot \underbrace{\begin{pmatrix} \frac{1}{g(\alpha_1)} & 0 & \dots & 0 \\ 0 & \frac{1}{g(\alpha_2)} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \frac{1}{g(\alpha_n)} \end{pmatrix}}_{\mathcal{Z}}$$
$$= \begin{pmatrix} \frac{1}{g(\alpha_1)} & \frac{1}{g(\alpha_2)} & \dots & \frac{1}{g(\alpha_n)} \\ \frac{\alpha_1}{g(\alpha_1)} & \frac{\alpha_2}{g(\alpha_2)} & \dots & \frac{\alpha_n}{g(\alpha_n)} \\ \vdots & \vdots & \dots & \vdots \\ \frac{\alpha_1^{t-1}}{g(\alpha_1)} & \frac{\alpha_2^{t-1}}{g(\alpha_2)} & \dots & \frac{\alpha_n^{t-1}}{g(\alpha_n)} \end{pmatrix} \quad (3)$$

This parity-check matrix form comes from the definition of an alternant code. Indeed, a Goppa code is a specific alternant code. Since $\mathcal{L}$ is assuming known, the matrix $\mathcal{Y}$ can be computed by an attacker. Let y_i be $1/g(\alpha_i)$ for all i for 1 to n. Then, by writing $\mathcal{H}$ over $\mathbb{F}_{2^m}$, $\mathcal{H}$ becomes for an attacker:

$$\mathcal{H} = \begin{pmatrix} y_1 & y_2 & \dots & y_n \\ \alpha_1 \cdot y_1 & \alpha_2 \cdot y_2 & \dots & \alpha_n \cdot y_n \\ \vdots & \vdots & \dots & \vdots \\ \alpha_1^{t-1} \cdot y_1 & \alpha_2^{t-1} \cdot y_2 & \dots & \alpha_n^{t-1} \cdot y_n \end{pmatrix},$$

where y_i are unknown for the adversary. This matrix leads to a linear system of tmn equations in $(tm)^2 + nm$ unknowns over $\mathbb{F}_2$. It was shown in [20, Section 5.6] that this attack succeeds only if the error weight is at most $\lfloor t/2 \rfloor$.

Another proposition was done in [15] using the definition of a generalized Reed-Solomon (GRS) code. Indeed, an alternant code is a specific GRS code, moreover a Goppa code is the restriction to $\mathbb{F}_2$ of a GRS code over $\mathbb{F}_{2^m}$. That is why both propositions are based on the same idea.

In addition, a Goppa code generated with g^2 instead of g, when g is irreducible, is still the same code [21, Chapter 12]: $\Gamma(\mathcal{L}, g) = \Gamma(\mathcal{L}, g^2)$. Therefore, using a parity-check matrix with g^2 instead of g, an attacker can get a linear system of twice more equations and hope to succeed in all error weight cases (up to t).

To finish about g recovering through a linear system given by a parity-check matrix, using the syndrome polynomial (1), another parity-check matrix of $\Gamma(\mathcal{L}, g)$ is:

$$\mathcal{H} = \begin{pmatrix} \frac{g_t}{g(\alpha_1)} & \frac{g_t}{g(\alpha_2)} & \cdots & \frac{g_t}{g(\alpha_n)} \\ \frac{g_{t-1}+\alpha_1 g_t}{g(\alpha_1)} & \frac{g_{t-1}+\alpha_2 g_t}{g(\alpha_2)} & \cdots & \frac{g_{t-1}+\alpha_n g_t}{g(\alpha_n)} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\sum_{i=1}^{t} \alpha_1^{i-1} g_i}{g(\alpha_1)} & \frac{\sum_{i=1}^{t} \alpha_2^{i-1} g_i}{g(\alpha_2)} & \cdots & \frac{\sum_{i=1}^{t} \alpha_n^{i-1} g_i}{g(\alpha_n)} \end{pmatrix}$$

$$= \underbrace{\begin{pmatrix} g_t & 0 & \cdots & 0 \\ & \ddots & & \vdots \\ g_{t-1} & g_t & \ddots & \\ \vdots & \ddots & \ddots & 0 \\ g_1 & \cdots & g_{t-1} & g_t \end{pmatrix}}_{\mathcal{X}} \cdot \underbrace{\begin{pmatrix} \frac{1}{g(\alpha_1)} & \frac{1}{g(\alpha_2)} & \cdots & \frac{1}{g(\alpha_n)} \\ \frac{\alpha_1}{g(\alpha_1)} & \frac{\alpha_2}{g(\alpha_2)} & \cdots & \frac{\alpha_n}{g(\alpha_n)} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\alpha_1^{t-1}}{g(\alpha_1)} & \frac{\alpha_2^{t-1}}{g(\alpha_2)} & \cdots & \frac{\alpha_n^{t-1}}{g(\alpha_n)} \end{pmatrix}}_{\mathcal{Y} \cdot \mathcal{Z}}$$

Since $g_t \neq 0$, $\mathcal{X}$ is invertible. Therefore, $\mathcal{X}\mathcal{Y}\mathcal{Z}$ and $\mathcal{Y}\mathcal{Z}$ define the same code. For further research, it can be interesting to try the attack proposed in [20] using these two matrices, with g and g^2, then compare all results.

Finally, another idea to recover the Goppa polynomial is to compute the Greatest Common Divisor (GCD) between all polynomials $f'_C(x)$ given in (5), with C as rows in $\mathcal{S} \cdot \mathcal{G}$. Since each row in $\mathcal{S} \cdot \mathcal{G}$ is a codeword and from [21, Chapter 12] we have:

$$C \in \Gamma(\mathcal{L}, g) \Leftrightarrow g(x) | f'_C(x), \tag{4}$$

where

$$\begin{aligned} f_C(x) &= \prod_{i=1}^{n} C_i (x - \alpha_i) \\ f'_C(x) &= \sum_{i=1}^{n} C_i \prod_{\substack{j=1 \\ j \neq i}}^{n} (x - \alpha_j), \end{aligned} \tag{5}$$

and with

$$\mathcal{S}_C(x) = \sum_{i=1}^{n} \frac{C_i}{x \oplus \alpha_i} = \frac{f'_C(x)}{f_C(x)}. \tag{6}$$

So the second way seems to be simpler, because there are $(k-1)$ GCDs to compute in the worst case instead of a linear system to solve. All together, a revelation of permutation matrix $\mathcal{P}$ is a significant step towards breaking the whole McEliece cryptosystem.

C. Measurement Setup

We attack software implementation of the McEliece PKC decryption algorithm running on a low-cost STM32F103 microcontroller (MCU) [22]. The MCU features a 32-bit ARM Cortex-M3 (clocked at 72 MHz) and 64kB Flash ROM (sufficient for McEliece decryption). We acquire traces by measuring voltage drop on a 1Ω resistor placed in series between a grounding pin of the MCU and the ground. The setup was especially developed for SCAs and it provides clear view on the MCU leakage.

Traces are acquired using the Agilent Technologies oscilloscope DSO9404A [23] (using 2 of 4 analog channels). All traces required for an attacking protected device are acquired at sample rate 250×10^6 samples per second (250 MS/s). Two 500 MHz passive probes were connected directly to *SubMiniature version A* (SMA) connectors available on the testing board with the MCU. Data acquisition is controlled by the software running on the oscilloscope's internal PC. The software sets up the oscilloscope, sends the ciphertext to the MCU – design under test (DUT) via UART and waits for the acknowledgment. The DUT rises a trigger and starts the ciphertext decryption. The oscilloscope measures power consumption during the first two steps of the decryption. Once the acquisition is finished, the PC stores the measured trace to the hard disk. The measurement process is repeated depending on the desired number of traces (usually from dozens to millions of traces, in our case it is 500).

D. Results of the CBDPA attack

We successfully deployed the CBDPA attack on a permutation matrix $\mathcal{P}^{-1}$. The attack is performed using power consumption traces with different and random inputs. Then, we perform the correlation analysis n times for each input bit - 1024 times in our case. We obtain positions of permuted bits by searching for correlation peaks during analyses as depicted in Fig. 2. By comparing positions of permuted bits with bits in input ciphertexts, we reconstruct the permutation matrix $\mathcal{P}^{-1}$.

During the attack, we measure 500 traces each with 1024 bits long random input ciphertexts $\tilde{C}$. As we can see in Fig. 2, we are able to find permuted positions of the input bits. Compared to the attack in [16], this attack produces a thin correlation peak as shown in Fig. 2. It is caused by a difference between implementations of the algorithms.We are able to extract the permutation matrix $\mathcal{P}$ with 100% success rate (1024×1024 bits). The proposed attacks can be enhanced to any value of n in order to attack other more practical implementations ($n = 2048, n = 4096, \ldots$).

IV. Novel CBDPA Countermeasure

At the end of [16], we outline a possible countermeasure. We further investigate and develop the idea in this paper. We introduce a novel implementation of the countermeasure (Algorithm 2). We are able to use the masking with decreased effects of two main disadvantages - requirement of random bits and increased computational complexity. Our new method of countermeasure uses the following properties of linear codes:

- rows in generator matrix $\mathcal{G}$ are codewords,
- every codeword has a zero syndrome,

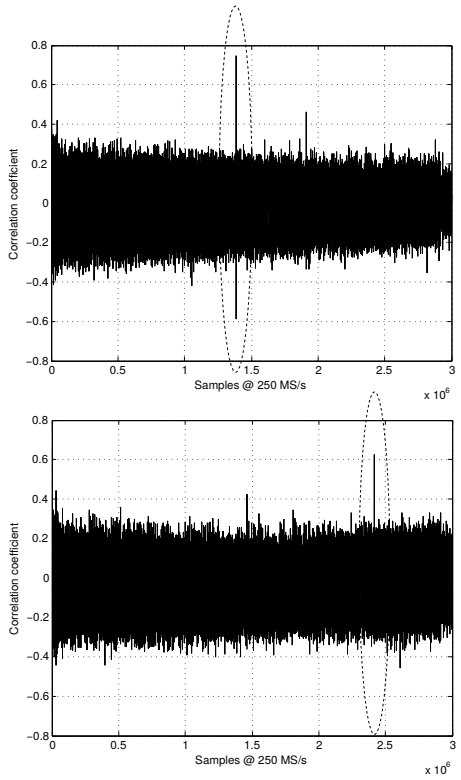


Figure 2. Correlation analyses using 500 power consumption traces. We can clearly distinguish the moment (marked with dashed ellipses) when the first bit (upper figure) and the second bit (lower figure) from input ciphertext are handled during the decryption algorithm (Algorithm 1). The first and the second bits are permuted to position 479 and 857 of 1024, respectively.

- syndrome computation is scheduled right after bit permutation in the McEliece decryption.

Algorithm 2 CBDPA resistant bit permutation and a syndrome computation (multiplication by the parity-check matrix) $S = [(\tilde{C} \oplus {}_pB) \cdot \mathcal{P}^{-1}] \cdot \mathcal{H}^T$

Require: Private permutation matrix $\mathcal{P}^{-1}$ represented by lookup-table $t^{\mathcal{P}}$, private parity-check matrix $\mathcal{H}^T$, ciphertext vector $\tilde{C}$, masks M_1, M_2 and a private generator matrix of $\Gamma(\mathcal{L}, g)$.
Ensure: Syndrome S
1: Choose ${}_pB$ such that ${}_pB = B \cdot \mathcal{P}$ and $B \in \Gamma(\mathcal{L}, g)$
2: $\tilde{C}' = \tilde{C} \oplus {}_pB$
3: $\tilde{C}'_p = M_1$
4: **for** $i = 0$ **to** $n-1$ **do**
5: $\quad tmp = unsigned(0 - \tilde{C}'_i)$
6: $\quad \tilde{C}'_p = \tilde{C}'_p \oplus (tmp \,\&\, (1 \ll t^{\mathcal{P}}_i))$
7: **end for**
8: $\tilde{C}'_p = \tilde{C}'_p \oplus M_1$
9: $S = M_2$
10: **for** $i = 0$ **to** $n-1$ **do**
11: $\quad tmp = unsigned(0 - \tilde{C}'_{p_i})$
12: $\quad S = S \oplus (tmp \,\&\, H^T_i)$
13: **end for**
14: $S = S \oplus M_2$
15: **return** S

Since the syndrome computation is a linear operation and the syndrome equals zero for all codewords, the main idea for our countermeasure is to add a codeword to the permuted ciphertext and compute the syndrome with this new word $\tilde{C}'_p = \tilde{C}_p \oplus B$, where $B \in \Gamma(\mathcal{L}, g)$ (for example one row in $\mathcal{G}$). The syndrome is the same since $S = \tilde{C}'_p \cdot \mathcal{H}^T = (\tilde{C}_p \oplus B) \cdot \mathcal{H}^T = \tilde{C}_p \cdot \mathcal{H}^T \oplus \underbrace{B \cdot \mathcal{H}^T}_{=0} = \tilde{C}_p \cdot \mathcal{H}^T$.

Hence we need to add the mask to the input ciphertext $\tilde{C}$ *before* the permutation algorithm, we add a permuted codeword ${}_pB$ such that ${}_pB \cdot \mathcal{P}^{-1} = B$. After the permutation, we get $\tilde{C}'_p = (\tilde{C} \oplus {}_pB) \cdot \mathcal{P}^{-1} = (\tilde{C} \cdot \mathcal{P}^{-1}) \oplus ({}_pB \cdot \mathcal{P}^{-1}) = \tilde{C}_p \oplus B$ which is input for the syndrome computation in the previous paragraph. From the point of view of an adversary, we add input ciphertexts to random values (with operation $\oplus$). The adversary can not know the variable tmp of the state 9 in Algorithm 1. The reason is that the hypotheses which the adversary would create are completely altered by the addition of the mask ${}_pB$ as shown in Fig 3.

In regular masking method, we need to generate random masks with width at least equal to the width of masked data. Afterwards, we perform calculations with mask and masked data. In the end, we merge mask and masked data into resulting data of the desired operation. In our case, we have to obtain a permuted Goppa codeword ${}_pB$. There are several possibilities to do so. It can be done either by choosing a random Goppa codeword or we can use a row in the generator matrix $\mathcal{G}$ (or linear combinations of rows) and modify them offline using the permutation matrix $\mathcal{P}^{-1}$. Another possibility is to use linear combinations of rows from public matrix $\tilde{\mathcal{G}}$ (this option is potentially vulnerable to an attack since an adversary knows the $\tilde{\mathcal{G}}$ matrix). For resource-constrained devices we can precompute a sufficient amount of permuted codewords ${}_pB$ and use their linear combinations for each decryption. After we add ${}_pB$ to the ciphertext, we get correct result from the McEliece PKC decryption *with no additional steps* assuming that the input ciphertext $\tilde{C}$ is stored in the MCU memory during the decryption process.

V. Experimental Results

The results of the new countermeasure in terms of computational complexity are in Tab. I. Compared to masking scheme, our novel method is more effective in terms of computational time and required random bits (RB). Using properties of the McEliece PKC, we avoid storing multiple values (masks and masked data) and we compute operations of permutation and parity-check matrix multiplication only once. Relations between intermediate values and input data are broken and thus the overall first-order DPA leakage is significantly reduced.

Results of the proposed CBDPA attack resistant bit permutation are given in and Fig. 4. We apply the novel countermeasure for the implementation of the permutation and then we repeat the CBDPA attack. If we compare the figures before and after the countermeasure, the results show that the leakage is significantly reduced. The CBDPA attacks are no longer possible as they were introduced.

VI. Conclusion

In this paper, we successfully deployed the CBDPA attack targeting the permutation matrix $\mathcal{P}$ of the McEliece PKC. The

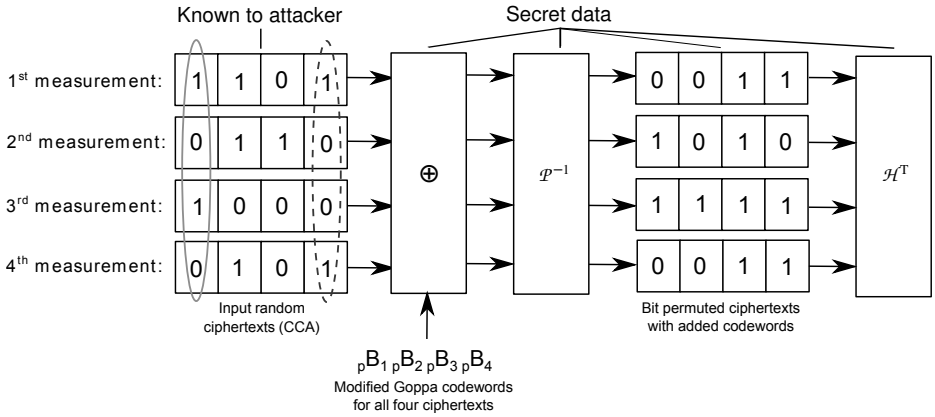


Figure 3. Secure bit permutation with the proposed countermeasure. Permuted Goppa codes are added to ciphertexts as masks before the permutation. During and after the permutation, an adversary is unable to find patterns (gray and dashed ellipses) in permuted ciphertexts with added Goppa codewords.

Secret bit permutation	Cycles	RB
SPA protected [15]	517,845	0
DPA prot. (our method)	518,211	0*
DPA prot. (regular masking)	1,036,422	n
SPA prot. [6], [16]	62,375,424	0
DPA prot. (our method)	62,375,790	0*
DPA prot. (regular masking)	124,751,580	n

Table I. COMPARISON TABLE BETWEEN VARIOUS IMPLEMENTATIONS OF A SECRET BIT PERMUTATION. THE CYCLE COUNTS ARE OBTAINED FROM 32-BIT ARM CORTEX-M3 MICROPROCESSOR. WE PERFORM ALL MEASUREMENTS WITH $n = 1024$ FOR EACH METHOD. SIMILAR RESULTS ARE OBTAINED FOR A SYNDROME COMPUTATION. *OUR METHOD CAN BE APPLIED WITHOUT RANDOM BITS WHEN A SUFFICIENT AMOUNT OF MASKING CODEWORDS IS PRECOMPUTED AND STORED IN ROM WITH USAGE OF PSEUDO-RANDOM BITS.

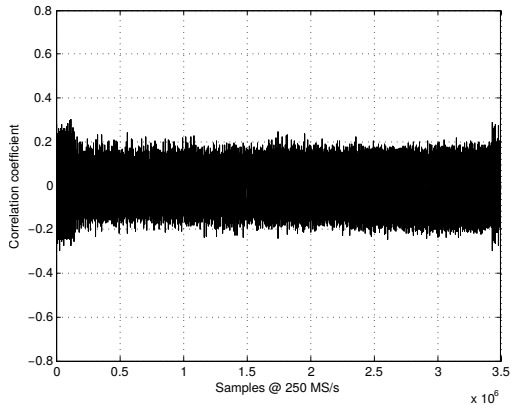


Figure 4. Correlation analysis using 500 power consumption traces. The figure corresponds to Fig. 2 with applied countermeasure. We are unable to detect significant correlation peaks in any of the 1024 correlation analyses.

algorithm was attacked on the ARM Cortex-M3 based MCU. We showed that we were able to fully recover the whole $n \times n$ permutation matrix. The attack can be extended to McEliece PKC parameters which are more suitable regarding current level of security, e.g. $n = 2048$.

Secondly, we develop and test an efficient countermeasure against possible CBDPA attacks on the implementation of the first steps of the McEliece PKC decryption. The countermeasure is based on a masking technique. We added Goppa codewords (random or rows of $\mathcal{G}$) to ciphertexts during the decryption algorithm as masks. By using properties of the McEliece PKC, we reduced the number of random bits and computational time that are required for the regular masking technique. After the codeword addition, there are no changes in the decryption algorithm thus the complexity of the countermeasure is significantly reduced.

Furthermore, we successfully measured and tested the proposed countermeasure. The results showed that the leakage of both implementations are significantly reduced. We did not intend to fully secure the algorithms with the novel countermeasure but we reduce the side-channel leakage with a method suitable for low-cost embedded and resource-constrained devices. Indeed, a higher order DPA attack is still possible as well as higher order masking techniques [24].

ACKNOWLEDGMENT

This work was performed in the framework of the COST Action IC1204 (Trustworthy Manufacturing and Utilization of Secure Devices). It was supported by the Slovak Research and Development Agency, project number APVV-0586-11 and in part by NATO's Public Diplomacy Division in the framework of 'Science for Peace', SPS Project 984520. Furthermore, the authors would like to thank Pascal Véron for his helpful advices.

REFERENCES

[1] R. Roman, P. Najera, and J. Lopez, "Securing the internet of things," *Computer*, vol. 44, no. 9, pp. 51–58, 2011.

[2] D. J. Bernstein, Y.-A. Chang, C.-M. Cheng, L.-P. Chou, N. Heninger, T. Lange, and N. van Someren, *Factoring RSA Keys from Certified Smart Cards: Coppersmith in the Wild*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 341–360. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-42045-0_18

[3] C.-C. Lee, C.-T. Li, C.-Y. Weng, J.-J. Jheng, X.-Q. Zhang, and Y.-R. Zhu, "Cryptanalysis and improvement of an ECC-based password authentication scheme using smart cards," in *Cyberspace Safety and Security*, ser. Lecture Notes in Computer Science, G. Wang, I. Ray, D. Feng, and M. Rajarajan, Eds., 2013, vol. 8300.

[4] D. J. Bernstein, J. Buchmann, and E. Dahmen, *Post Quantum Cryptography*, 1st ed. Springer Publishing Company, Incorporated, 2008.

[5] R. J. McEliece, "A public-key cryptosystem based on algebraic coding theory," California Inst. Technol., Pasadena, CA, Tech. Rep. 44, January 1978.

[6] F. Strenzke, E. Tews, H. G. Molter, R. Overbeck, and A. Shoufan, "Side channels in the McEliece PKC," in *The Second International Workshop on Post-Quantum Cryptography (PQCrypto 2008)*, ser. Lecture Notes in Computer Science, J. Buchmann and J. Ding, Eds. Berlin Heidelberg: Springer, October 2008, vol. 5299, no. 5299/2008, pp. 216–229. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-88403-3_15

[7] N. J. Patterson, "The algebraic decoding of Goppa codes," *IEEE Transactions on Information Theory*, vol. 21, no. 2, pp. 203–207, March 1975.

[8] A. Shoufan, F. Strenzke, H. G. Molter, and M. Stöttinger, "A timing attack against Patterson algorithm in the McEliece PKC," in *Proceedings of the 12th International Conference on Information, Security and Cryptology (ICISC 2009)*, ser. Lecture Notes in Computer Science, D. Lee and S. Hong, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 5984, pp. 161–175. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-14423-3_12

[9] F. Strenzke, "A timing attack against the secret permutation in the McEliece PKC," in *Proceedings of the Third international conference on Post-Quantum Cryptography (PQCrypto 2010)*, ser. Lecture Notes in Computer Science, N. Sendrier, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 6061, pp. 95–107. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12929-2_8

[10] R. M. Avanzi, S. Hoerder, D. Page, and M. Tunstall, "Side-channel attacks on the McEliece and Niederreiter public-key cryptosystems," *Journal of Cryptographic Engineering*, vol. 1, no. 4, pp. 271–281, November 2011. [Online]. Available: http://dx.doi.org/10.1007/s13389-011-0024-9

[11] F. Strenzke, "Timing attacks against the syndrome inversion in code-based cryptosystems," in *The 5th International Workshop on Post-Quantum Cryptography (PQCrypto 2013)*, ser. Lecture Notes in Computer Science, P. Gaborit, Ed. Berlin Heidelberg: Springer, 2013, vol. 7932, pp. 217–230. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38616-9_15

[12] S. Heyse, A. Moradi, and C. Paar, "Practical power analysis attacks on software implementations of McEliece," in *Proceedings of the Third international conference on Post-Quantum Cryptography (PQCrypto 2010)*, ser. Lecture Notes in Computer Science, N. Sendrier, Ed. Berlin Heidelberg: Springer, 2010, vol. 6061, pp. 108–125. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12929-2_9

[13] H. G. Molter, M. Stöttinger, A. Shoufan, and F. Strenzke, "A simple power analysis attack on a McEliece cryptoprocessor," *Journal of Cryptographic Engineering*, vol. 1, no. 1, pp. 29–36, April 2011. [Online]. Available: http://dx.doi.org/10.1007/s13389-011-0001-3

[14] C. Chen, T. Eisenbarth, I. von Maurich, and R. Steinwandt, "Differential power analysis of a McEliece cryptosystem," Cryptology ePrint Archive, Report 2014/534, 2014. [Online]. Available: http://eprint.iacr.org/2014/534

[15] M. Petrvalsky, T. Richmond, M. Drutarovsky, P.-L. Cayrel, and V. Fischer, "Countermeasure against the spa attack on an embedded mceliece cryptosystem," in *Radioelektronika (RADIOELEKTRONIKA), 2015 25th International Conference*, April 2015, pp. 462–466.

[16] M. Petrvalsky, T. Richmond, M. Drutarovsky, P. L. Cayrel, and V. Fischer, "Differential power analysis attack on the secure bit permutation in the mceliece cryptosystem," in *2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA)*, April 2016, pp. 132–137.

[17] V. D. Goppa, "A new class of linear error-correcting codes," *Problemy Peredachi Informatsii*, vol. 6, no. 3, pp. 24–30, September 1970.

[18] E. R. Berlekamp, "Goppa codes," *IEEE Transactions on Information Theory*, vol. 19, no. 5, pp. 590–592, September 1973.

[19] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *CHES*, 2004, pp. 16–29.

[20] R. Heiman, "On the security of cryptosystems based on linear error correcting codes," Master's thesis, Feinburg Craduate School of the Weitzmann Institute of Science, Rehovot, August 1987.

[21] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*, N.-H. M. Library, Ed. North-Holland, 2006.

[22] ST Microelectronics, "STM32 product information, software and datasheets," http://www.st.com/web/en/catalog/mmc/FM141/SC1169, [online] Cited 12/12/2015.

[23] Agilent Technologies, "DSO9404A datasheet and product information," http://www.keysight.com/en/pd-1632456-pn-DSO9404A/oscilloscope-4-ghz-4-analog-channels?&cc=SK&lc=eng, [online] Cited 12/12/2015.

[24] M. Rivain, E. Prouff, and J. Doget, "Higher-order masking and shuffling for software implementations of block ciphers." in *CHES*, ser. Lecture Notes in Computer Science, C. Clavier and K. Gaj, Eds., vol. 5747. Springer, 2009, pp. 171–188.

Influence of Fault-Tolerant Design Techniques on Resistance against Differential Power Analysis

Vojtěch Miškovský

Czech Technical University in Prague
Faculty of Information Technology
Department of Digital Design

CryptArchi 2017
June 18–21, Smolenice

Motivation

- Reliability and security are often demanded at the same time
- Both fault-tolerant and attack-resistant design techniques usually have high area and power consumption overhead, that means even higher overhead when both are used
- The overhead of fault-tolerant design can influence the attack-resistance of the device
- The overhead of attack-resistant design can influence the reliability of the device

Long-term goals

- Examine the mutual influence of fault-tolerant and attack-resistant digital design architectures
- Discover some interplay between F-T and A-R design and use it to design both F-T and A-R architecture with lower summary overhead

Current work

- Only the side-channel attacks are related to digital design
- We started with the most common one — Differential power analysis (DPA)
- We experimentally evaluate how basic fault-tolerant architectures affects the resistance against DPA
- FPGA implementation of AES is used

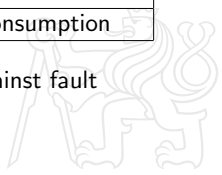
Related work

Similar study was presented by Regazzoni et al. [1]

Table: Comparison of key features of Regazzoni et al. approach and our approach

Regazzoni	Our Approach
Fault attack resistant design	Fault-tolerant design
S-Box protected	the whole encryptor protected
ASIC	FPGA
Simulated power consumption	Real power consumption

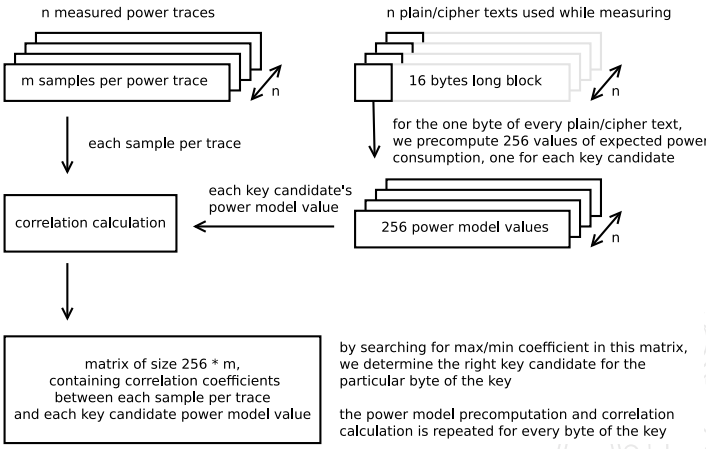
Regazzoni concluded that the designs protected against fault attacks are more vulnerable to power attacks.



Differential Power Analysis

- Side-channel attack, introduced by Kocher et al. [2]
- Exploits the fact that a variable in the implementation exists, that
  - it depends on the plain/cipher text used,
  - it depends on a part of a key,
  - and it correlates with the power consumption of the device
- Correlation power analysis is an enhanced variant of DPA, attacking one byte of a key at a time, analyzing correlation between:
  - measured power consumption samples
  - and estimated power consumption model, based on plain/cipher text and a one of the key candidates

Differential Power Analysis



by searching for max/min coefficient in this matrix, we determine the right key candidate for the particular byte of the key

the power model precomputation and correlation calculation is repeated for every byte of the key

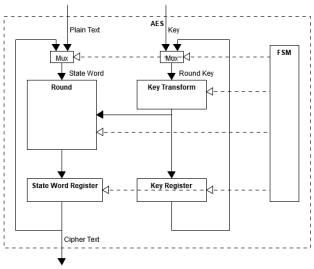
F-T architectures

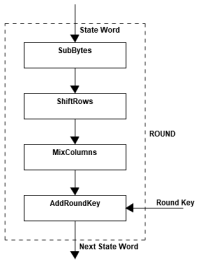
These fault-tolerant architectures were used:

- Information redundancy — S-Box Parity check (AES-SPC)
- Space redundancy
  - round level (AES-HR-R)
  - algorithm level (AES-HR-A)
- Time redundancy
  - round level (AES-TR-R)
  - algorithm level (AES-TR-A)

Plain AES implementation

128-bit variant, 10 rounds
11 clock cycle per encryption (1 initial round + 10 rounds)
After each round the state word is stored in a register

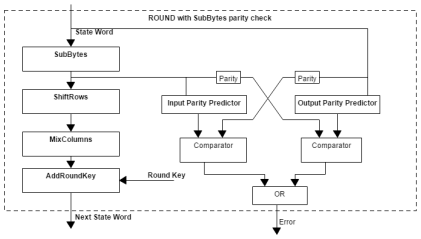




Information redundancy

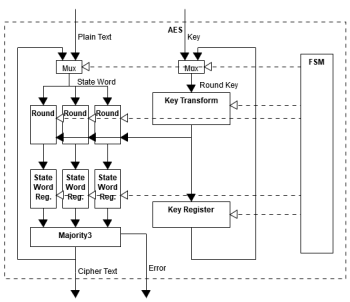
S-Box (SubBytes function) is secured by parity
Two parity predictors are used:

- Input parity predictor
- Output parity predictor



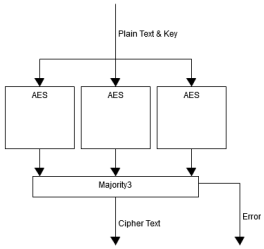
Space redundancy — Round level

The round is protected by TMR (3 copies and a majority voter)



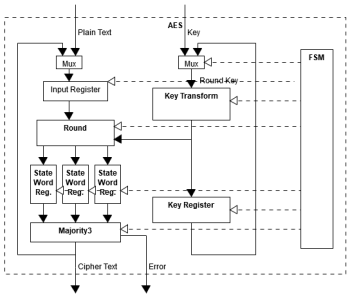
Space redundancy — Algorithm level

The whole AES is protected by TMR (3 copies and a majority voter)



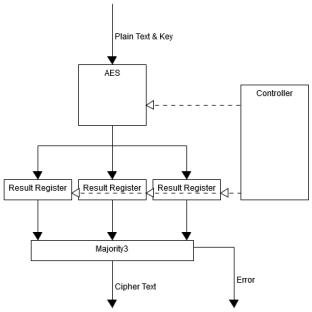
Time redundancy — Round level

Each round is repeated 3 times and the results are compared by a majority voter
Only works for transient faults



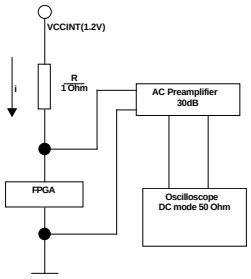
Time redundancy — Algorithm level

The whole algorithm is repeated 3 times and the results are compared by a majority voter
Only works for transient faults



Measurement — Hardware

Evariste II board with Altera Cyclone III FPGA module was chosen as an implementation platform [3]
Power consumption was measured by PicoScope 6404D



Measurement — Software

- We developed our own application for DPA
- The correlation calculations are time and memory efficient and highly numerically stable
- Qualities of the application were published at DDECS 2017 [4]
- Now we work on GPU implementation
- This application is planned to be extended to a whole DPA framework and published under open-source license

Experiment setup

- We obtained 50 sets of 2000 powertraces for each F-T variant of AES, 1000 samples per powertrace
- For each set, we performed the DPA with various number of powertraces to find the minimal numbers of powertraces sufficient to obtain the correct cipher key (*minTraces*)
- The AES variants are compared by medians of the *minTraces*
- Lower *minTraces* means lower resistance against DPA (it is easier to perform the attack)

Results

Table: Comparison of AES variants based on median and interquartile range of *minTraces*

Architecture	Median	Interquartile range	Diff. from AES
AES	850	175	0%
AES-SPC	950	250	+12%
AES-HR-R	900	275	+6%
AES-HR-A	812	150	-4%
AES-TR-R	1025	250	+21%
AES-TR-A	1037	275	+22%

Results

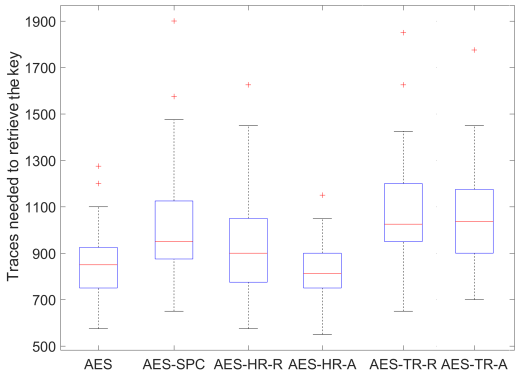


Figure: Box plot of *minTraces* of measured AES architectures

Conclusion

- We show that basic redundancy techniques have insignificant influence on resistance against DPA on implementation not secured against DPA
- We presume these techniques can be used to make DPA resistant devices fault-tolerant
- This statement should be verified on another FPGA platforms
- Our results are contrary to results presented by Regazzoni et al. We believe that the main reason is the different power consumption evaluation (simulation vs. real measurement)

References

[1] F. Regazzoni, L. Breveglieri, P. Ienne, and I. Koren, "Interaction between fault attack countermeasures and the resistance against power analysis attacks," in *Fault Analysis in Cryptography*. Springer, 2012, pp. 257–272.

[2] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Annual International Cryptology Conference*. Springer, 1999, pp. 388–397.

[3] V. Fischer, F. Bernard, and P. Haddad, "An open-source multi-FPGA modular system for fair benchmarking of true random number generators," in *Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on*. IEEE, 2013, pp. 1–4.

[4] P. Socha, V. Miškovský, H. Kubátová, and M. Novotný, "Optimization of Pearson correlation coefficient calculation for DPA and comparison of different approaches," in *2017 IEEE 20th International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS)*, April 2017, pp. 184–189.

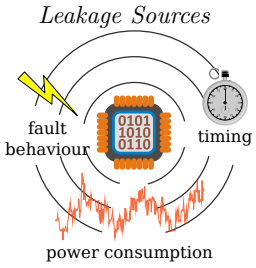
The Design-Time Side-Channel Information Leakage Estimation

Jan Bělohoubek
jan.belohoubek@fit.cvut.cz

Faculty of Information Technology
Czech Technical University in Prague

Motivation
Information Leakage

- Digital circuits offer sensitive information while computation(side-channel)
- Today circuit designers compete with attackers:
  - Designers are trying to build circuits resistant to SPA, DPA, Fault-attack, Combined (Fault + PA) ...
  - → **Decrease** the information offered thru side-channel
  - → **Measure** the information offered thru side-channel



- **Unbalanced data/control paths** (Different loads, Place&Route, Early evaluation)
- **Unbalanced computation (data-dependent algorithms)**
- Completion detection – asynchronous circuits

Localize Weakness and Estimate Potential

- **How to distinguish good idea[1] and bad idea during the different design phases?**
  - post-Synthesis – what can be achieved with current design?
  - post-Map – what can be achieved with current cell library?
  - post-Place&Route – how will behave the physical design?

How to measure vulnerability?

- **Production time**
  - Number of traces needed to break the circuit (get AES key)
- **Design time**
  - Use number of traces [2] – accurate simulation + many traces → time !?
  - Use well established methods – make conservative (but subjective) estimation → accuracy !?
  - **Do we have objective and efficient metric?**

The Method

Using Power Traces

- **The sensitive information leaking from the circuit influences the character of the power traces**
  - Timing - differential peak position; duration of the computation
  - Fault - differential peak position, width or height; duration of the computation
  - Unbalanced paths - differential peak position, width or height

→ Many types of information leakage are aggregated in power traces

→ **Using only power traces for vulnerability evaluation is sufficient**

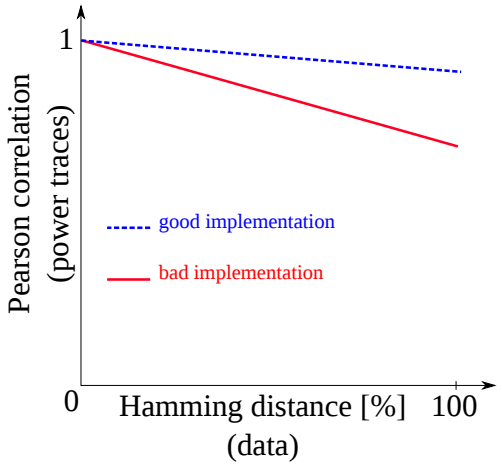
[1] Is a certain circuit implementation better from the side-channel vulnerability point of view?

[2] K. Smith and M. Łukowiak, "Methodology for simulated power analysis attacks on AES," 2010 - MILCOM 2010 MILITARY COMMUNICATIONS CONFERENCE, San Jose, CA, 2010, pp. 1292-1297.

- **What is Required?**
  - Fast vulnerability estimation allowing incorporation into the design flow process
  - Measure the information contained in power trace
  - Estimation at different design levels – post-Synthesis, post-Map, post-Place&Route
- **What is Observed?**
  - The information in the power trace is proportional to the similarity of traces
    - → if all traces would be equal, the attacker can extract no information
    - → if there is a dependency between the processed data and power trace patterns, the attacker may extract information

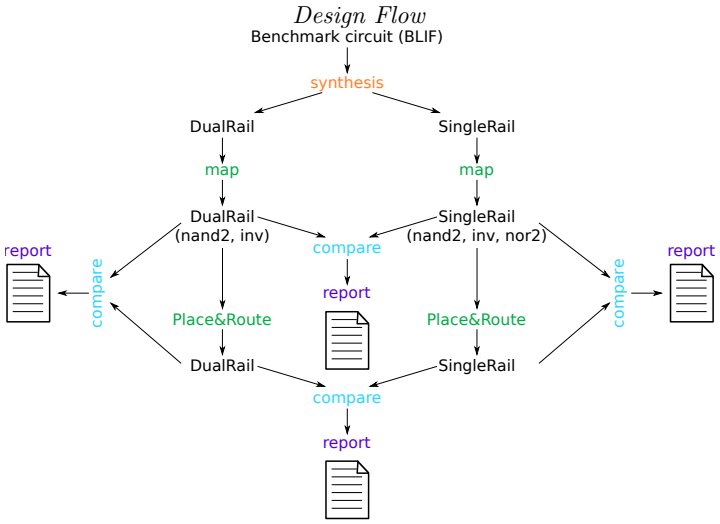
Data vs. Power Trace Dependency

- **Let's search for data vs. power trace dependency**
  - Data similarity metric: **Hamming distance**
  - Power trace similarity metric: **Pearson correlation**
  - → **Is correlation of traces for similar data high and for different data (significantly) low?**



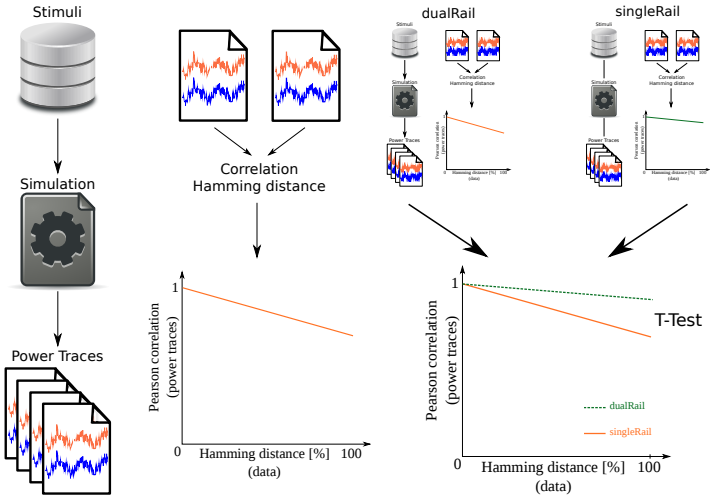
Methodology
The Current Design Potential

- post-Synthesis – what can be achieved with current design?
  - No physical layer information!
  - Is simulation-based estimation possible? **It is not possible without any assumption about technology!**
- post-Map – what can be achieved with current cells?
  - Take information about cells only (parasitic capacitances, conductivity, . . .)
  - Interconnection is assumed ideally balanced (or zero delay/power)
  - Place&Route can make things worse
- post-Place&Route – the "reality"
  - Should be close to physical design



- Generate the stimuli set:
  - initial vector is generated randomly
  - other vectors are derived by inverting bits in the initial vector

  → the stimuli set contains vectors with Hamming distances (0% – 100 %)
- Use stimuli to get power traces (simulation)
- Compute Pearson correlation for all pairs of power traces
- Build a data-set containing pairs: [Hamming distance, Correlation] (plot ...)
- Compare different implementations: formulate hypothesis and test by using the t-test



Simulation

Tools

- Spice – open (ngSpice); too accurate; **too slow**
- Synopsys PrimeTime PX – commercial – looks fine (not tested yet)
- IRSIM – open alternative to PTPX?; fast; **too old**
  - Produces event times, not power traces (poweEst package is available)
  - Good for CMOS with lambda ≥ 1 μm technology
  - For CMOS below 1 μm, the results looks bad – characterization failed

  ...

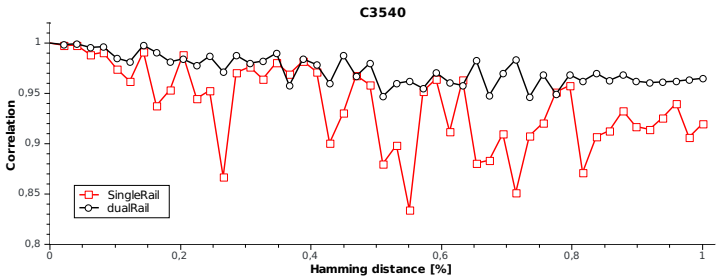
Combinational Circuits

- stimuli set contains i vectors, where i is equal to # of circuit inputs

$\rightarrow$ we have $i^2/2$ pairs of vectors with all possible Hamming distances

- number of stimuli vectors is reduced
- SPICE simulation is feasible for relative small circuits like C3540:
  - $\approx$ 1000 gates
  - 50 inputs
  - 1250 input vector and power trace pairs

IRSIM – Above 1 μm

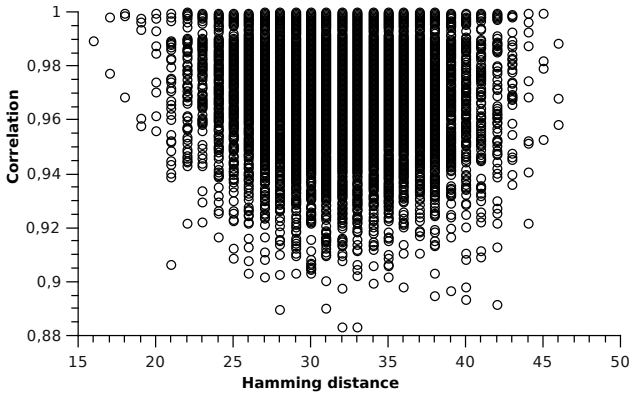


\+ Nice graph, looks as expected – T-test (and my eyes) says: **singleRail is** (much) **worse than dualRail**

– IRSIM gives similar results for TSMC180nm – here **disagrees with SPICE!** (wrong tech. characterization)

Measurements

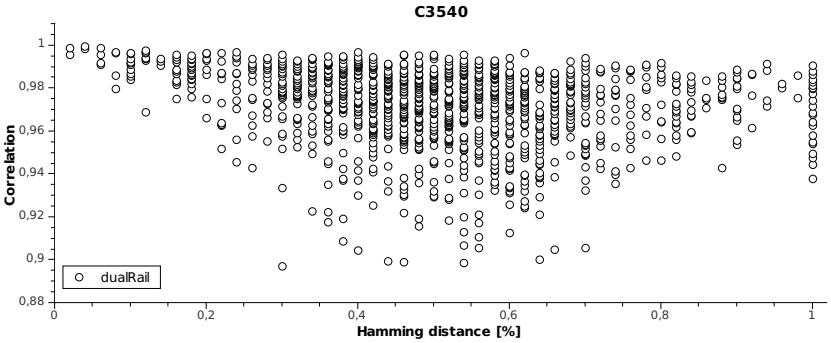
3 years ago . . . CryptArchi 2014



- Real measurements – Asynchronous dualRail DES on FPGA

Simulation

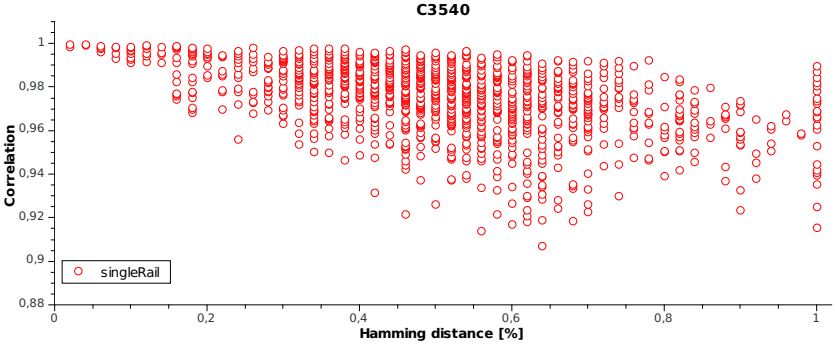
SPICE – DualRail



- dualRail layout (TSMC180nm) of the benchmark circuit C3540

\+ precise SPICE simulation looks very similar to measured data! (C3540 is similar to DES)

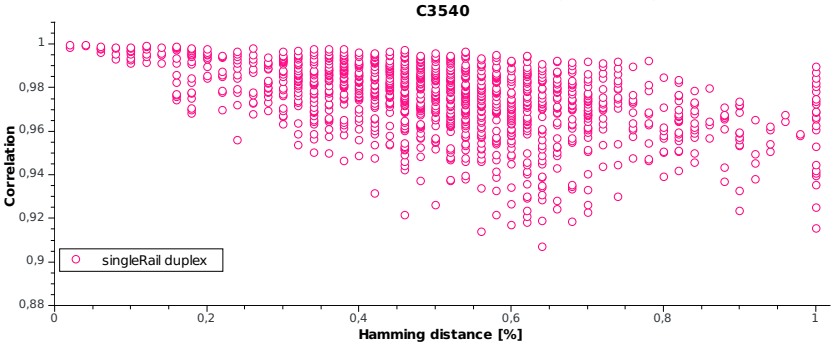
SPICE – SingleRail



– SingleRail has less variation and the minimum of singleRail is above dualRail

– T-Test (not my eyes here!) says: **singleRail is better!** (a bit)

SPICE – Duplex of SingleRails (no voter)



- The sum of two singleRails is equal to the single SingleRail – **no additional information leakage!**

Summary

Preliminary Results Show Interesting Facts

When no manufacturing variations were taken into account:

1 More logic working data-dependently is bad → information leakage is increased
   - both branches of DualRail circuits perform data-dependent computations → **balancing becomes extremely important!**

2 Adding more logic blocks producing exactly the same power traces is OK → NMR will not increase information leakage

When manufacturing variations will be taken into account, the 2. case will slightly became case 1!

Future Work and Challanges

- Is it possible to measure information leakage simpler?

→ the area of circuit parts performing data-dependent computations independently

- Is singleRail really better than dualRail in practice? ... No!

→ Where are the limits of masking (balancing dual rails)?

→ What is the relationship of information leakage and circuit vulnerability?

→ Is the attacker's strength estimation – without focusing to the particular attack – possible?

- There is no (open) efficient and accurate simulator of CMOS producing power traces.

Highlights

- The information leakage is proportional to the amount of logic working data-dependently!
- The presented method is able to estimate information leakage (fast open simulator is missing).
- Ideal duplex (no voters!) does not offer additional information to attacker.

Acknowladgements

This research has been partially supported by the grant GA16-05179S of the Czech Grant Agency and by CTU grant SGS17/213/OHK3/3T/18. Computational resources were provided by the CESNET LM2015042 and the CERIT Scientific Cloud LM2015085, provided under the programme "Projects of Large Research, Development, and Innovations Infrastructures".

CRYPTARCHI 2017 Smolenice "Formalism to Assess the Entropy and Reliability of Loop PUF"

Jean-Luc Danger[1,2]
Olivier Rioul[1]
Sylvain Guilley[1,2]
Alexander Schaub[1]

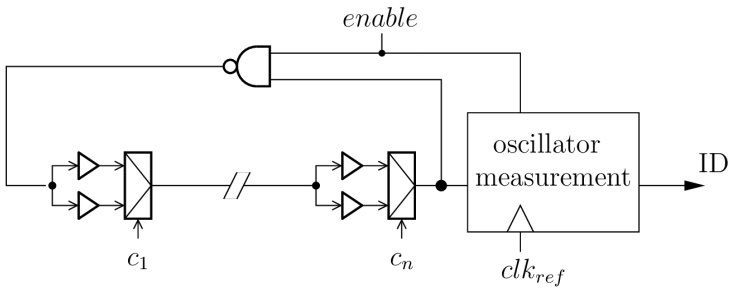
[1] Télécom ParisTech, LTCI, UPSAY
[2] Secure-IC

Outline

- Loop PUF architecture
- Entropy assessment
- Reliability assessment
- Results on real silicon
- Conclusions

* Depends on algorithm, not implementation

Loop PUF architecture



Operating Mode

Algorithm 2.1 Operating Mode with 2 complementary challenges

Input: Challenge C (a word of n bits)
Output: Response B

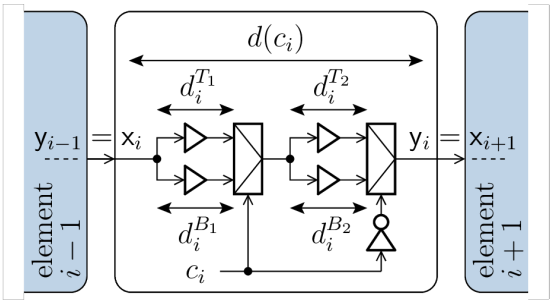
1: Set challenge C
2: Measure $d_C \leftarrow \lfloor L \sum_{i=1}^{n} d(c_i) \rfloor$
3: Set challenge $\neg C$
4: Measure $d_{\neg C} \leftarrow \lfloor L \sum_{i=1}^{n} d(\neg c_i)$
5: Compute $\Delta = d_C - d_{\neg C}$
6: Return $B = \text{sign}(\Delta) \in \{\pm 1\}$

The information for each challenge is Δ

Sign = identifier

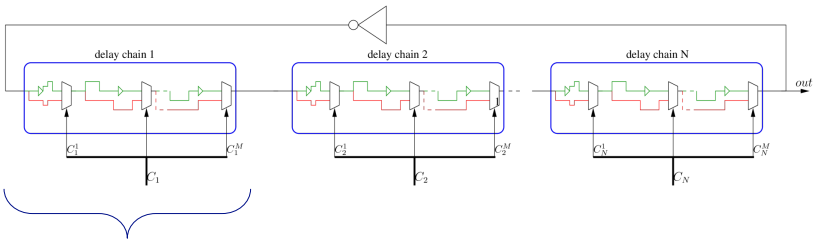
Module = reliability

Balance of Delay Elements in ASIC



duplication

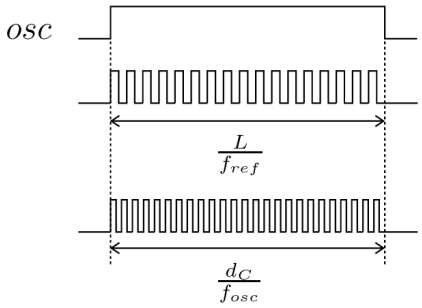
Balance of Delay Elements in FPGA



Cluster 1 Cluster 2 Cluster N

duplication N duplications

Delay measurement



$$d_C = L \cdot \frac{f_{osc}}{f_{ref}}$$

Entropy

- **For a n-delay LPUF**
  - If challenge = Hadamard codeword of n bits => Entropy = n *

For instance for $n = 12$, the n by n Hadamard Matrix is:

$$\mathcal{C}_{12} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 1 \\ 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 \end{pmatrix}$$

* Rioul, O., Solé, P., Guilley, S., & Danger, J. L. (2016, July). On the Entropy of Physically Unclonable Functions. In *Information Theory (ISIT), 2016 IEEE International Symposium on* (pp. 2928-2932). IEEE.

Entropy with more than n challenges

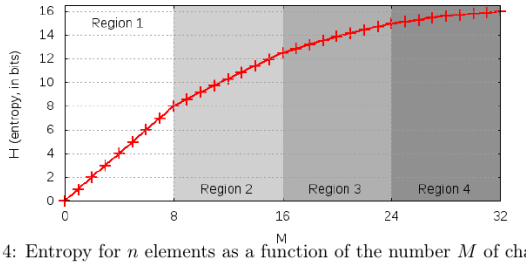


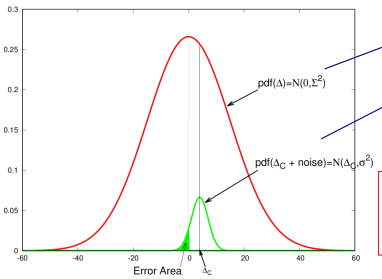
Figure 4: Entropy for n elements as a function of the number M of challenges.

LPUF reliability

$$BER = \mathbb{P}(\text{sign}(\Delta + N) \neq \text{sign}(\Delta)) = Q(\frac{|\Delta|}{\sigma})$$

$$N =\sim \mathcal{N}(0, \hat{\sigma}^2) \qquad Q(x) = \tfrac{1}{2}(1 - \text{erf}\,\tfrac{x}{\sqrt{2}}) = \tfrac{1}{2}\,\text{erfc}(\tfrac{x}{\sqrt{2}})$$

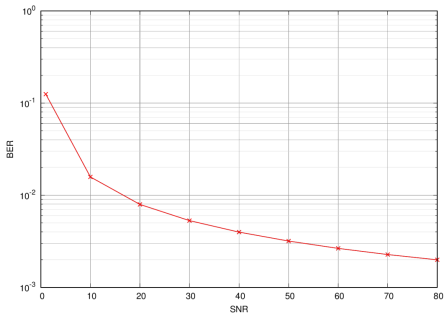
$$\widehat{BER} = \int_0^{+\infty} \mathbb{P}(\Delta) \cdot BER(\Delta) \cdot d\Delta$$



With $SNR = \frac{\Sigma}{\sigma}$

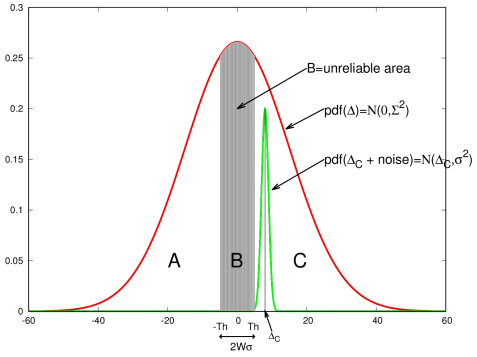
$$\widehat{BER} = \frac{1}{4} - \frac{1}{2\pi}\arctan(SNR)$$

LPUF reliability



The Reliability is not enough ~10^{-3} even with high SNR
=> Needs of secure sketch : Error Correcting codes and Helper data

Reliability enhancement by delay knowledge



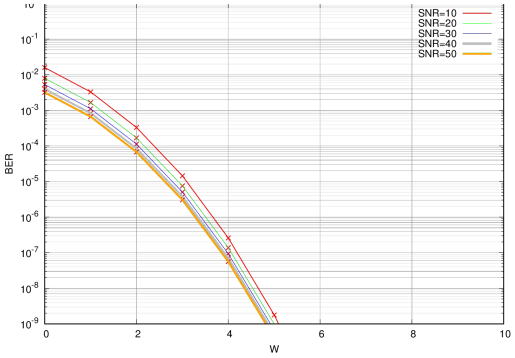
Bit unreliable ⇔
|delay| < Th

Th= Wσ

The bits in the unreliable area "B" are discarded
The helper data indicates the unreliable bits

New BER with filtered bits

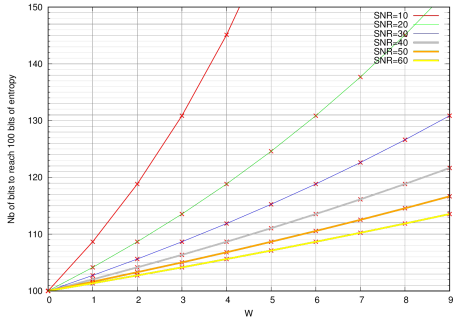
$$\widehat{BER_{filt}} = T(W, \frac{1}{SNR}) + \frac{1}{4}\,\mathrm{erf}(\frac{W}{\sqrt{2}\cdot SNR})(\mathrm{erf}(\frac{W}{\sqrt{2}}) - 1)$$



Entropy after bit filtering

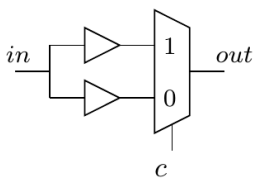
Number of delay elements to reach n bits of entropy with Hadamard codes

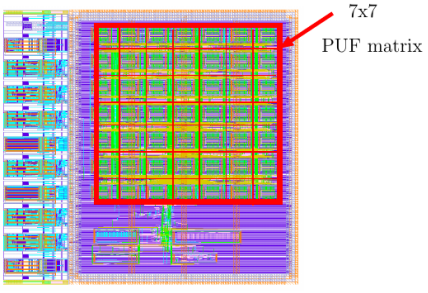
$$n' = \frac{n}{1 - \mathbb{P}(Bit\ unreliable)} = \frac{n}{\mathrm{erfc}(\frac{W}{SNR})}$$



Results on real silicon

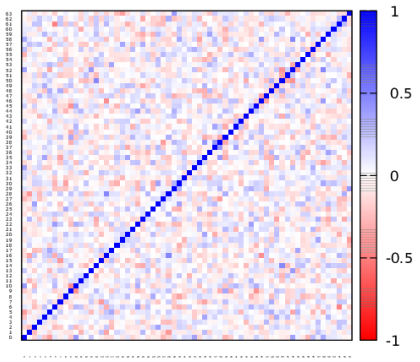
- **n=54 cells, 65nm technology**





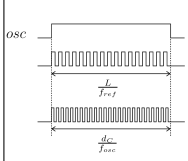
i..i.d. check

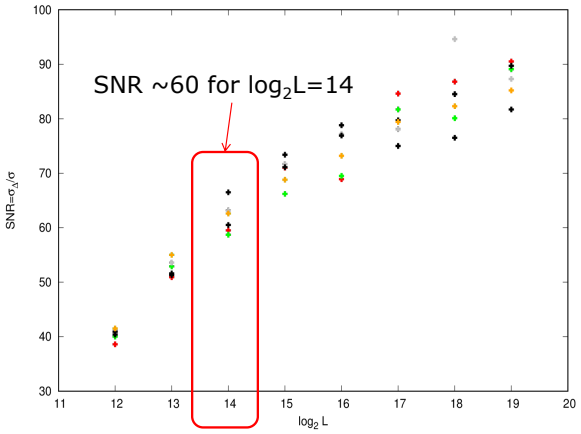
Correlation matrix on the 64 elements of the 49 PUFs



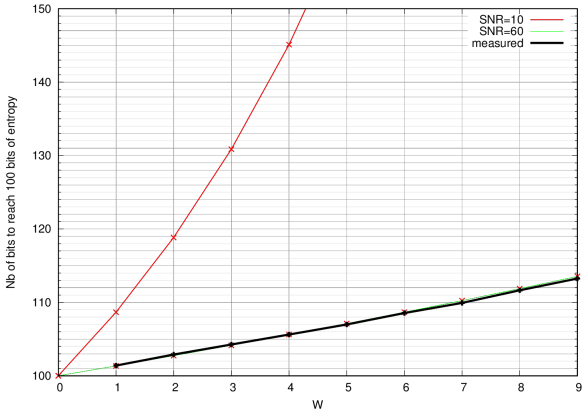
No correlation between the 64 delay elements
=> entropy ~ 64 with Hadamard codes

Impact of the measurement window on the SNR

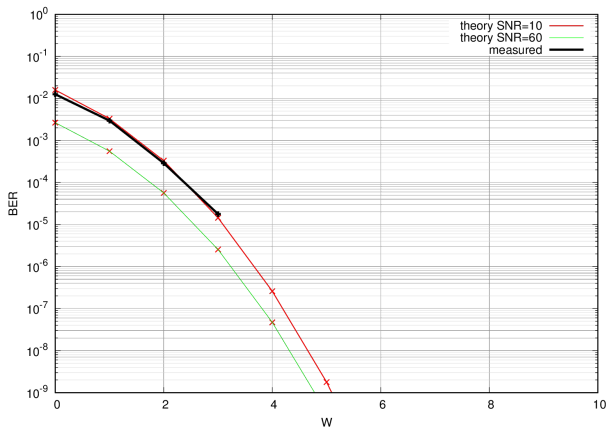




Entropy



Reliability: BER results



Conclusions

- **The Entropy of the Loop PUF can be formally obtained if Hadamard codes are used:**
  - Entropy = number of delay elements n
  - The entropy increases non linearly if M > n

N=Number of challenges

- **The reliability of the Loop PUF is low (BER ~10^{-3})**
- **It can be easily improved by exploiting the delay knowledge**
  - The unreliable bits are discarded
  - BER can go down 10^{-9}
  - But more bits are needed to reach the same entropy

THANK YOU
FOR YOUR ATTENTION

Clone-Resistant Structures in Microsemi SoC Units

Cryptarchi 2017

18-21 June 2017
Smolenice Castle, Slovakia

W. Adi, A. Mars

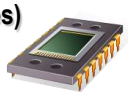
IDA, Institute of Computer and Network Engineering
Technical University of Braunschweig
Germany

Contents

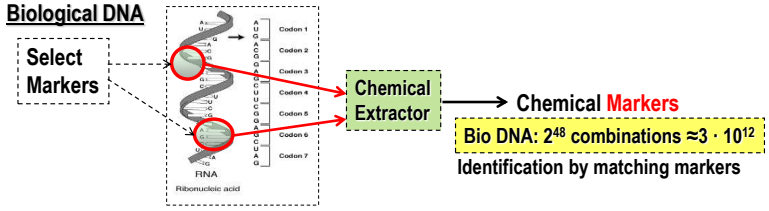
Why clone-resistant physical units?

- Commercial-economic reasons (Cloning)
- Identity (Privacy)
- Know-How protection (IP-Cores)
- Medical
- Automotive units
- E-Money ....
- Smart-Home, -City, -Gouvernement, Consumer, IoT ..




Best physical Identity: As the born DNA-like provable identification



Physical Unclonable Functions PUFs offer DNA like Identification Techniques

Ideal PUFs are: Born unpredictable and unclonable physical VLSI properties.

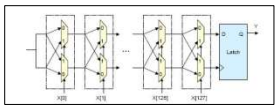
In other words: PUFs are analogue non-linear, hard to model or to copy, unpredictable huge mapping in a semiconductor VLSI device

State of the Art: Unclonable Devices by: Analog Physical Unclonable Functions (PUFs)

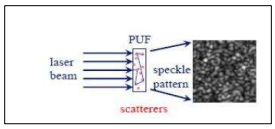
Since 2000 many proposals

- *optical PUF*
- *coating PUF*
- *Silicon PUF*
- *optical fiber PUF*
- *RF COA*
- *LC-PUF*
- *S-RAM PUF*
- *Arbiter PUF*
- *fluorescent PUF*

Unclonable DNA-Chain

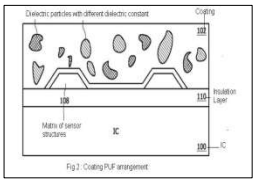


Delay based



Optical

- *Delay PUF*
- *Butterfly PUF*
- *diode breakdown PUF*
- *reconfigurable PUF*
- *acoustic PUF*
- *controlled PUF*
- *phosphor PUF*
- ...

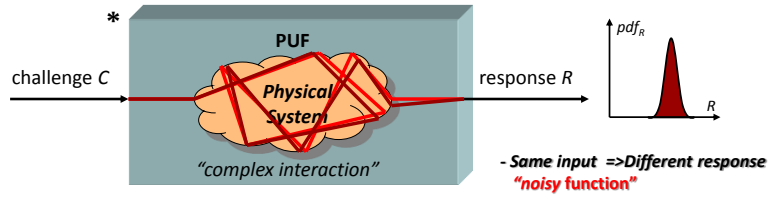


Coating PUF (Capacity)

So far all have "Reproducibility" problems! Analog functions!!!

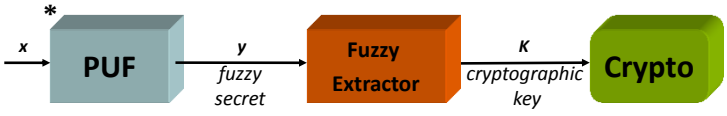
Born properties: Similar to the biological DNA

PUFs inconsistency and aging difficulties



- Same input =>Different response "noisy function"

Bad reproducibility due to : Operating conditions, quantization (Metastability) , Aging, ...



Fuzzy Extractors: Complex, Costly (Sign. Proc + ECC)

* Source: Roel Maes, ESAT/COSIC, K.U.Leuven, BCRYPT Workshop:

Our Proposal:

1. Avoid analog-world
   and use pure digital structures
   => 100% reproducibility, no aging!
   => Resilient/robust Identification Technology

2. Accept less strict security requirements
   for mass consumer products.

Less-strict security requirements

Pragmatic Low-Cost Practical Cloning-Security for mass products:

- System is considered as clone-resistant if:
  - Cloning do not economically pay off
  - Cloning is useless after some short time

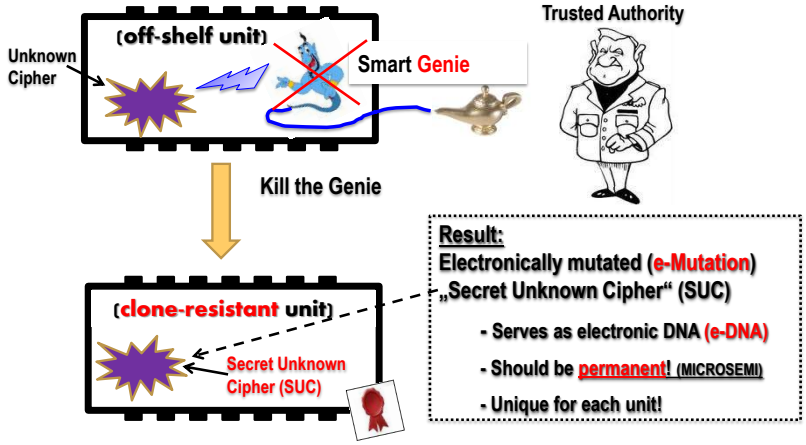
Therefore, Security Requirements

- Attacker enforced to do expensive physical invasive attacks
  ⇒ Side Channel attack resistant
- "Break-one break-all" should be hard or impossible
  ⇒ Each unit is uniquely individual

Target applications: Mass-products, Automotive .. Consumer, Smart phones

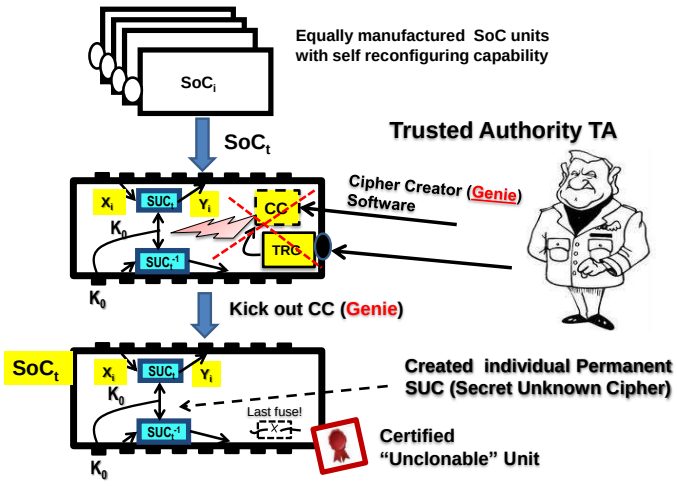
Key Idea: Electronic DNA-Mutation!

In Post Fabrication: "Mutating a digital "Secret Unknown Cipher" (SUC)



Production Procedure: Post fabrication

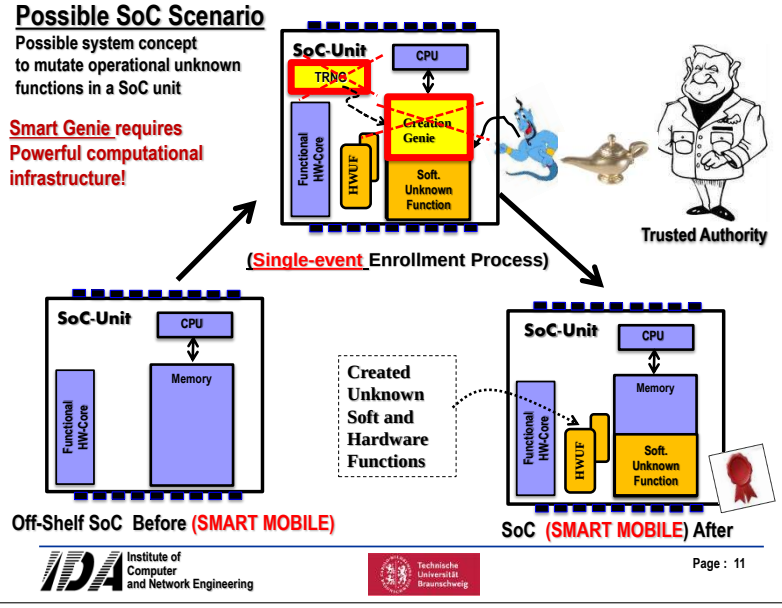
"Mutate SUC"



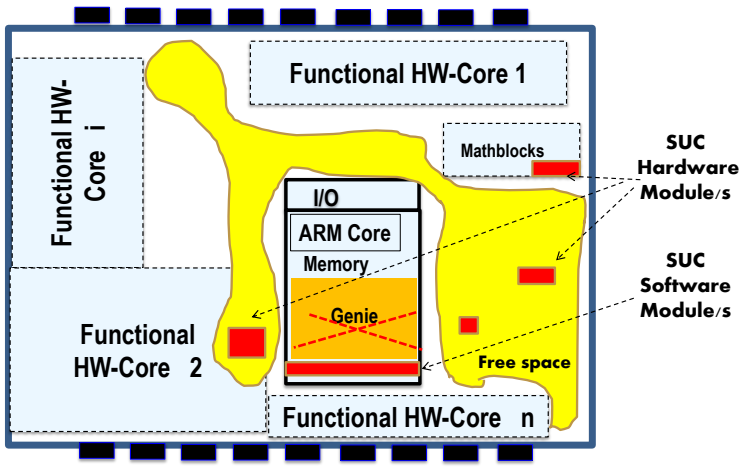
Possible SoC Scenario

Possible system concept to mutate operational unknown functions in a SoC unit

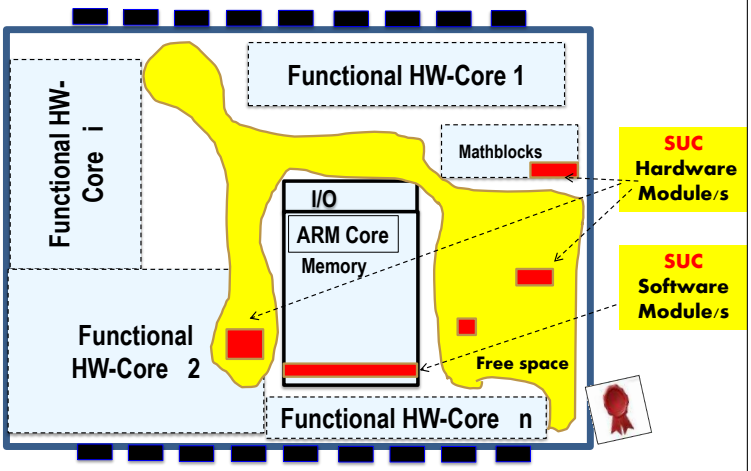
Smart Genie requires Powerful computational infrastructure!



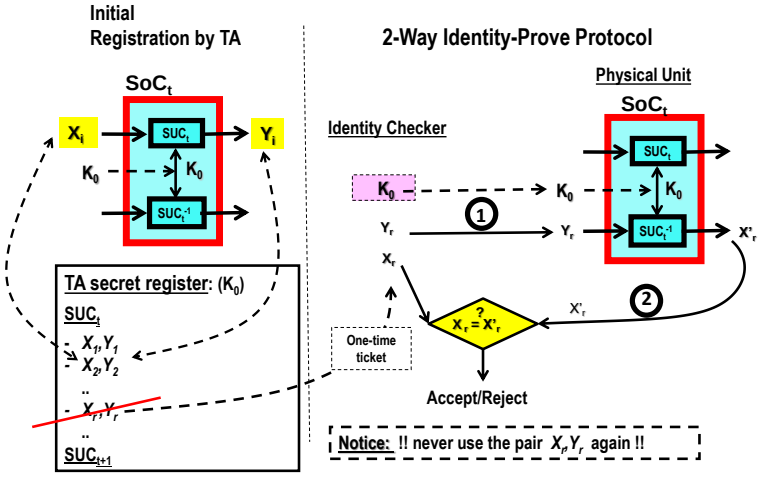
SUC Prototype in SmartFusion2 SoC FPGA



Clone-Resistant SoC FPGA Unit



Unit Identification Protocol with a SUC



Q: Is the system realizable today?

- In ultimate form , No! (Emerging .. Technology)

Reason: non-volatile self reconfiguring hardware in SoC architecture is not yet available

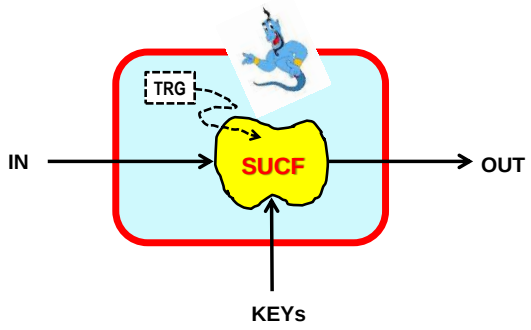
However: *"Microsemi non-Volatile technology"* (Actel) is a possible future emerging technology therefore

Pure Software solution?:
"ARM SoC" architecture is a possible infrastructure
However less secure, low-cost soft version (Smart phones)

Generalized Concept of Unknown Secret-key Crypto as Clone-Resistant Mutated Digital E-DNA Functions

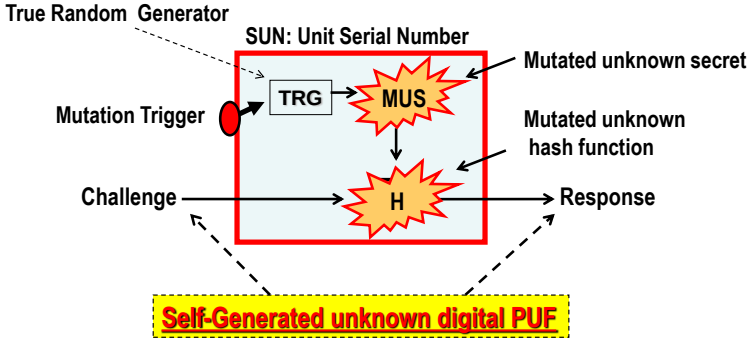
Created Functions

Secret Unknown Crypto-Functions" SUCF

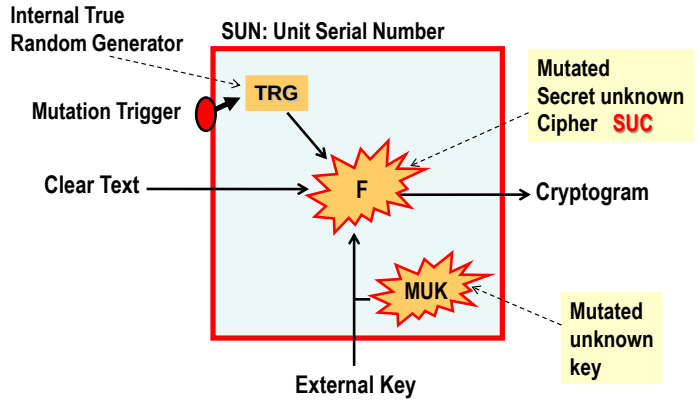


Use Case 1: Mutating Unknown Hash & key

Mutated secret identity



Use Case 2: Mutating Unknown Secret Physical Cipher



Sample Implementation Case

SUC in Microsemi SmartFusion®2 SoC FPGAs

Ultimate SUC
"is not Realizable"
in Contemporary Technologies!

Why?:

1. No self-reconfiguring non-Volatile Technologies
2. Bitstream Format is not disclosed
3. Bitstream is encrypted

First Pragmatic Implementation Scenario:

Concept:

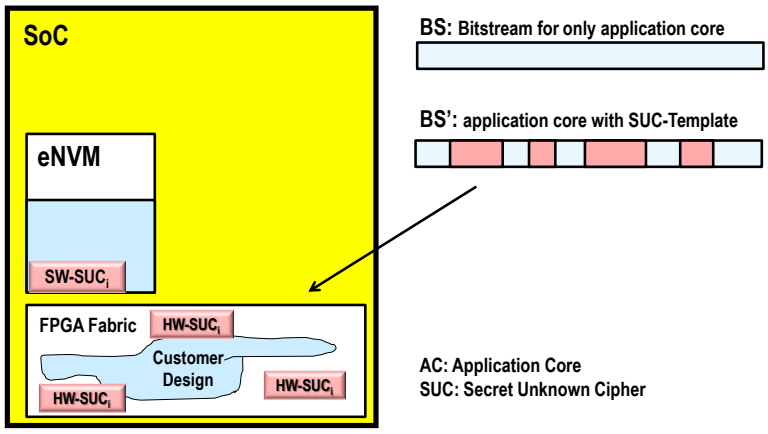
1. Use existing technologies
2. Accept lower security

Note:

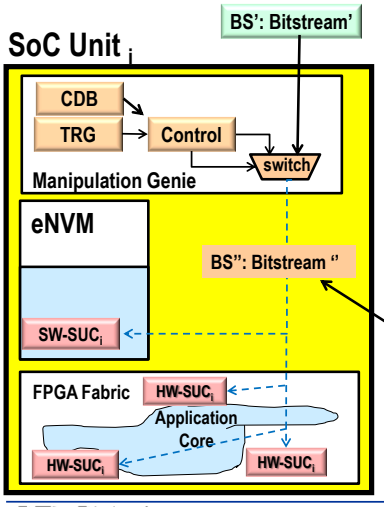
Self-Reconfiguring Volatile Technologies do exist, however result with very weak security for unclonability

Prototyping Key Idea:

Step-1: Creating SUC as a "Hardware Layout Template"



Prototyping Key Idea: Step-2: Template personalization



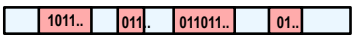
Initial setup:

- Create a SUC template in SoC,

Personalize Units by a Manipulation GENIE!:

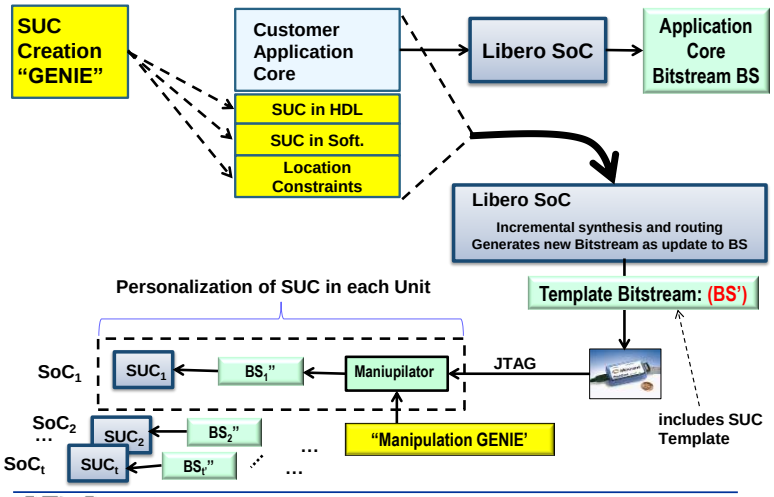
1. Trigger GENIE' by a random stream of bits from the TRG
2. Chose randomly some mappings from the cipher data base (CDB)
3. Manipulate the configuration bitstream BS' accordingly
4. After completing the SUC personalization, the GENIE' is deleted (killed)

BS'': Bitstream after personalization

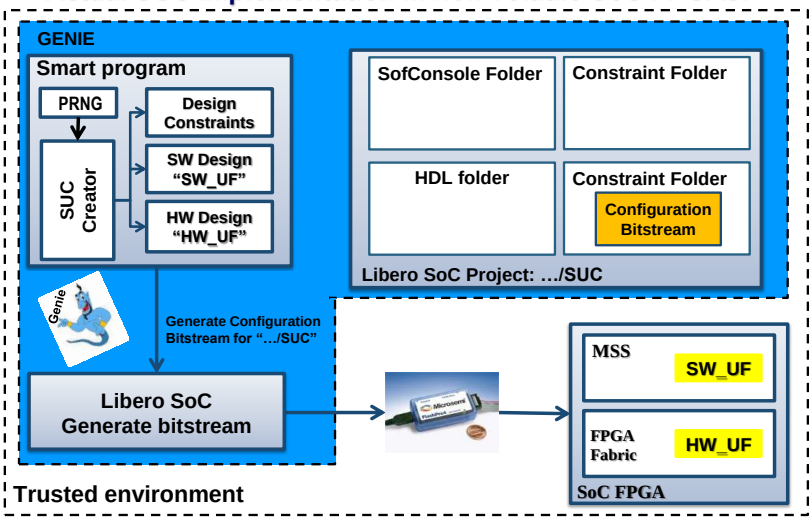


AC: Application Core
CDB: Cipher Data Base
TRG: True Random Generator
SUC: Secret Unknown Cipher

Design Flow for Creating SUC in Microsemi SoC FPGAs:



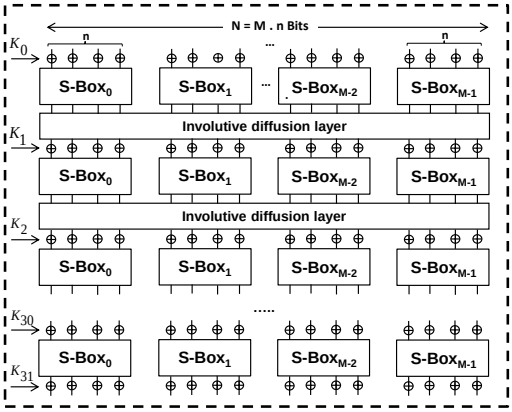
Actual SUC Implementation in non-volatile SoC FPGAs:



Proposed Randomized Involution Block Cipher as SUC:

Requirements: Secure design, lightweight, involutive, and random cipher

proposed SUC : has a block size N=64 bits and 32 rounds



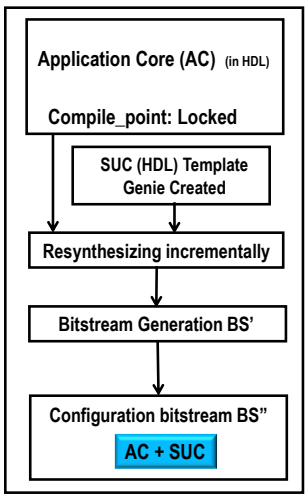
Cipher Cardinality:

- The cardinality of this class of random cipher is 2^{274}

Attack complexity:

- Linear cryptanalysis: $\geq 2^{80}$
- Differential cryptanalysis: $\geq 2^{80}$

Soft-Design flow: Incremental SUC Synthesis and Routing

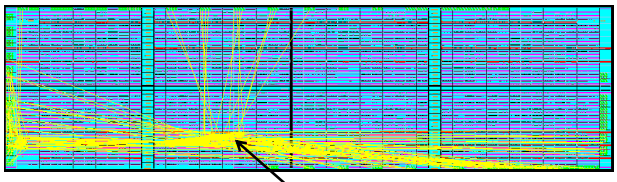


- Compile points are RTL partitions of the design that you define before synthesizing the design
- Each compile point is treated as a block
  → Independent synthesis, optimization and place and route

- The SUC design template can be added to locked Application Core,
- The software will treats separately the SUC design which will reduce the time required for personalization

Sample Layout: By incremental synthesis and routing

Complete layout



Pure SUC portions and their locations

Hardware complexity:

The following table presents the hardware complexity of SUC (in percent) for different SmartFusion®2 SoC FPGAs Families

Gate Complexity: 213 LUTs + 72 DFFs

SmartFusion2 SoC FPGA Families	*Resources usage*	
	LUTs % of usage	*DFFs % of usage*
M2S005	3,51	1,19
M2S010	1,76	0,6
M2S025	0,77	0,26
M2S050	0,37	0,12
M2S060	0,37	0,12
M2S090	0,24	0,08
M2S150	0,14	0,04

Summary of the First SUC Prototyping

Concept:

1. Template based
2. Fixed cipher architecture
3. Only mapping-contents are variable
4. Low Complexity. Very high personalization speed

Disadvantages:

1. Attacker knows the structure
2. Attacker knows the physical locations
3. No crypto-mappings diversity (fixed template!)

However,
overall security level is still relatively: "Very good"

CONCLUSIONS

- Relatively low-cost pure Digital PUFs (in best case "zero-cost")
- "Highly robust digital physical identity" (compared to analog PUFs !)
- Negligible aging!
- Scalable security level!.
- System inherently more resistant to „Side Channel Attacks"
- Security is, manufacturer and trusted authority independent

Work in Progress:

- Investigating new "GENIEs", operation scenarios and use protocols
- Practical real field applications,
- Task is multidisciplinary and challenging!

Thanks

HECTOR

Secure Portable USB Data Storage

HECTOR Demonstrator 2 platform

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644052

Marcel Kleja (1), Marek Laban (1,2), Viktor Fischer (3)
(1) MICRONIC a.s., Bratislava, Slovak Republic
(2) Technical University of Kosice, Slovak Republic
(3) Laboratoire Hubert Curien Saint-Etienne, France

Secure Portable Data Storage

- USB mass storage class flash drive with integrated HW cipher
- Designed to protect sensitive personal information
  - Protects data stored on it at rest
  - Target audience: lawyers, doctors, notaries...

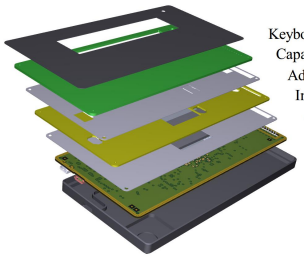
- Motivation for development of such a device:
  - Weaknesses have been discovered in similar secure solutions (key stored "securely" in the device)
  - Lack of trusted secure storage solutions engineered and manufactured in EU
  - To demonstrate HECTOR project outcomes

Secure Portable Data Storage

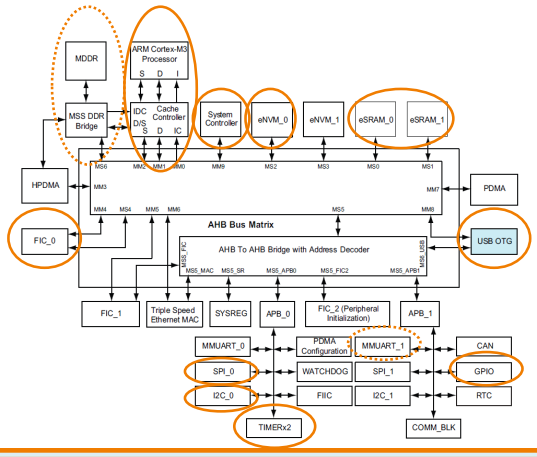
- Data are stored on a replaceable SD-card (up to 32 GB)
- Based on Microsemi SmartFusion2
- Passphrase entered directly on device – reduces risk of keylogger attacks
- The data throughput is >19MB/s for read and >13MB/s for write

Secure Portable Data Storage – HW Development

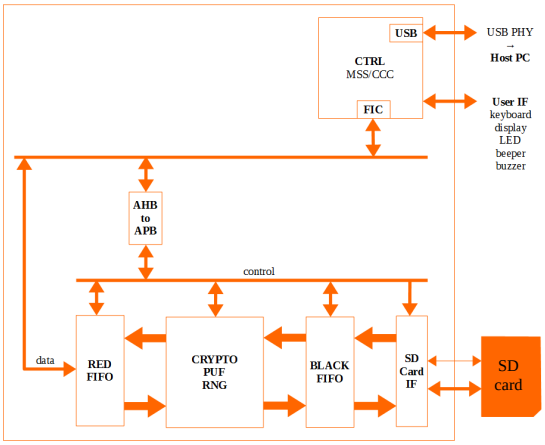
- Complex process – several versions developed
  - First version based on existing commercial evaluation boards
  - Second version on HECTOR evaluation boards
  - Final version:
    - Capacitive keyboard implemented
    - Custom case developed
    - Device is assembled using adhesive foils
    - Protected against disassembling by poured epoxy mass in the bottom case



Microsemi SmartFusion2 – MSS Utilization



Microsemi SmartFusion2 – FPGA Fabric



HECTOR Outcomes in Demonstrator

- Authenticated encryption
  - CAESAR competition candidate - ASCON used
- True Random Number generator
  - PLL-TRNG with integrated embedded tests compliant with AIS 20/31 PTG.2
- Physically unclonable function
  - TERO PUF with postprocessing (32-bits of the 128-bit key)

HECTOR Evaluation Platform

- Based on motherboard and daughter board modular system
- Developed for easy evaluation and development of cryptographic primitives
- Main advantages:
  - Low-cost, exchangeable daughter boards
  - Adapted to implementation of side-channel attacks
  - Support for daughter board remote connection via HDMI cable

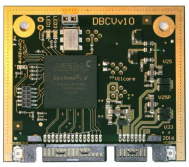
HECTOR Evaluation Platform – Motherboard

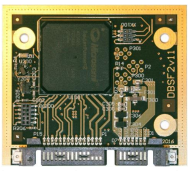
- Based on Microsemi SmartFusion2
- 64 MB of external RAM, USB interface
- Only low-noise linear regulators are used
- Complex acquisition system implemented
- Controlled by the PC using a USB interface and TCL scripts

Development Process

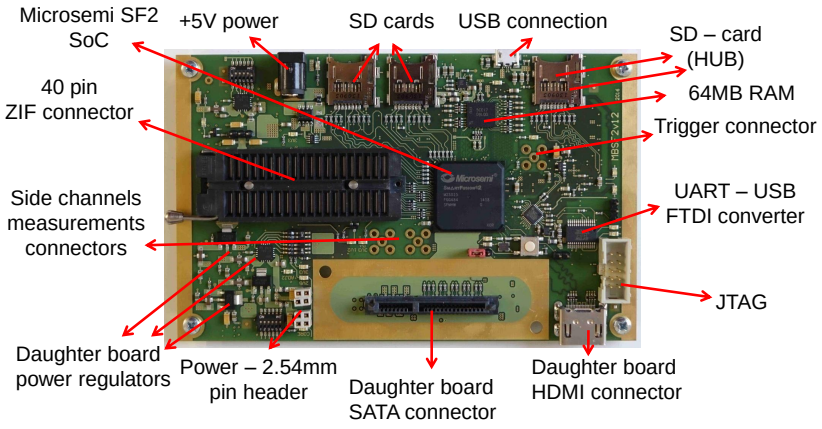
- Primitives examined on FPGAs of three different families
- Evaluated on many FPGAs in order to verify reproducibility and reliability
- Tested and measured in various environmental conditions using a remote connection of the module
- Evaluation platform – a step to demonstrator development


Xilinx Spartan 6

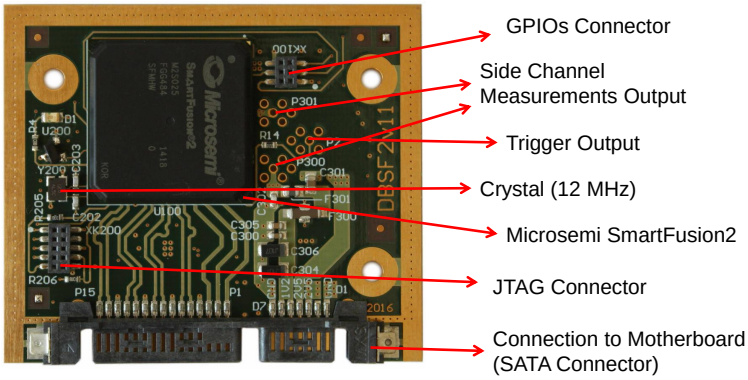

Altera Cyclone V


Microsemi SmartFusion 2

HECTOR Mother Board 1 – Microsemi SF2



HECTOR Microsemi SmartFusion 2 Daughter Board



HECTOR

Usage of the Device

- Empty
  - Security critical data zeroized
- Enrolled
  - Security critical data generated: helper data, data encryption key
- Once enrolled – user just enters his passphrase to generate 96 bits of the key (remaining 32 come from the PUF)

HECTOR

Demonstration

- Follows…

HECTOR

Conclusions

- The device demonstrates results of the HECTOR project
- It can be used as a secure HW storage of highly sensitive data
- Main advantages:
  - The confidential key is not stored in the device
  - A part of the key is device dependent but not reachable by the attacker (PUF)

"The **HECTOR** project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 644052."

If you need further information, please contact the coordinator:
TECHNIKON Forschungs- und Planungsgesellschaft mbH
Burgplatz 3a, 9500 Villach, AUSTRIA
Tel: +43 4242 233 55 Fax: +43 4242 233 55 77
E-Mail: coordination@hector-project.eu

Complete activation scheme for IP design protection

Brice Colombier[1], Ugo Mureddu[1], Marek Laban[2], Oto Petura[1], Lilian Bossuet[1], Viktor Fischer[1]
[1]Hubert Curien Laboratory, UMR CNRS 5516, University of Lyon, 42000 Saint-Étienne - France
{b.colombier, ugo.mureddu, oto.petura, lilian.bossuet, fischer}@univ-st-etienne.fr
[2]Department of Electronics and Multimedia Communications, Technical University of Košice, Park Komenskho 13
04120 Košice, Slovak Republic,
MICRONIC, Sliačska 2/C, 83102, Bratislava, Slovak Republic
laban@micronic.sk

***Abstract*—Intellectual Property (IP) illegal copying is a major threat in today's integrated circuits industry which is massively based on a design-and-reuse paradigm. In order to fight this threat, a designer must track how many times an IP has been instantiated. Moreover, illegal copies of an IP must be unusable. We propose a hardware/software scheme which allows a designer to remotely activate an IP with minimal area overhead. The software modifies the IP efficiently and can handle very large netlists. Unique identification of hardware instances is achieved by integrating a TERO-PUF along with a lightweight key reconciliation module. A cryptographic core guarantees security and triggers a logic locking/masking module which makes the IP unusable unless the correct encrypted activation word is applied. The hardware side is implemented on several FPGAs.**

I. Goal of the demo

The goal of the proposed hardware demo is to show how a designer can modify an IP so that it can be activated remotely and securely. Before activation, the IP can be instantiated but is unusable. Its outputs are either forced to a fixed logic value or disturbed. Later, upon activation request, the designer sends an encrypted activation word. This is then decrypted inside the IP to activate it. Each IP instance is made unique by integrating a PUF, leveraged to derive a secret key. It prevents a malicious system integrator from instantiating the IP on a non-trusted hardware target. We make the whole system open-source.

II. Experimental setup

From a hardware perspective, the experimental setup (Fig. 1) comprises a laptop and an FPGA board, connected via a serial interface. A user interface can perform the following actions:

- Modify the IP, using logic masking [1] or logic locking [2] to make it controllably unusable. Several parameters can be tuned, as well as the area overhead.
- Obtain the reference response from the TERO-PUF [3] during the enrolment phase.
- Reconcile the key with CASCADE [4] and activate the IP.

III. Demo scenario and observables

The typical demo scenario is the following. First, an IP in the form of a netlist is modified and the associated activation word (AW) is stored. The motherboard is then connected to the PC and the daughter-board is enrolled by obtaining a response from a PUF instantiated at a known location. This response is used to encrypt AW. The protected IP is instantiated on the enrolled daughter-board. Before activation, the IP does not operate correctly. When the activation phase starts, the key reconciliation procedure is conducted to ensure that the PUF response generated on the daughter-board is identical to the one obtained during enrollment. Then, AW is encrypted and sent to the board. It is then internally decrypted and sent to the logic masking/locking module, to make the IP fully operational. If the IP is instantiated on a different daughter-board, it does not operate correctly since the PUF response is different. Each IP is then securely bound to a trusted hardware target.

Acknowledgments

The work presented in this paper was realized in the frame of the SALWARE project number ANR-13-JS03-0003 supported by the French "Agence Nationale de la Recherche" and by the French "Fondation de Recherche pour l'Aéronautique et l'Espace", funding for this project was also provided by a grant from "La Région Rhône-Alpes".

This project has also received funding from the European Unions Horizon 2020 research and innovation program under grant agreement no. 644052.

References

[1] J. A. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending piracy of integrated circuits," in *DATE*, 2008, pp. 1069–1074.
[2] B. Colombier, L. Bossuet, and D. Hély, "Reversible denial-of-service by locking gates insertion for IP cores design protection," in *IEEE ISVLSI*, 2015, pp. 210–215.
[3] A. Cherkaoui, L. Bossuet, and C. Marchand, "Design, evaluation and optimization of physical unclonable functions based on transient effect ring oscillators," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1291–1305, 2016.
[4] B. Colombier, L. Bossuet, D. Hély, and V. Fischer, "Key reconciliation protocols for error correction of silicon PUF responses," *IACR Cryptology ePrint Archive*, vol. 2016, p. 939, 2016.

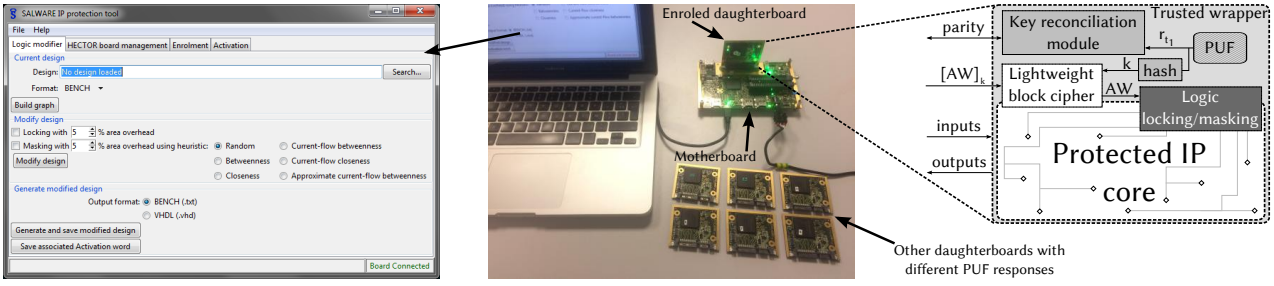


Fig. 1. Experimental setup showing the software user interface and the hardware wrapper added to the IP.

ALESSIO

The problem

- Successful automation solutions tend to be used over considerable stretches of time (> 10 years)
- Security requirements are hard to predict over larger periods of time
- Outdated security in older systems implies a considerable risk, e. g. for critical infrastructure
- **The only means to achieve durable security is updatability**

The problem understood by the German Federal Ministry of Education and Research

The German Federal Ministry of Education and Research partially funds research for updatable security in the project ALESSIO

- Infineon Technologies AG, Neubiberg (Prime contracor)
- Giesecke & Devrient GmbH, München
- Fraunhofer-Institut für Angewandte und Integrierte Sicherheit (AISEC), Garching bei München
- Technische Universität München
- WIBU-SYSTEMS AG, Karlsruhe
- Siemens AG, München

From January 2017 to December 2019
Total costs 6.8 M€, 58% federal funding

Siemens: 1 M€, 40% funding

What does the name ALESSIO stand for?

SIEMENS
Ingenuity for life

Architektur für langfristige Sicherheit
durch Secure Elements mit Update-Funktion

Architecture for long term security
by secure elements with update function

ALESSIO as a bipartite project

SIEMENS
Ingenuity for life

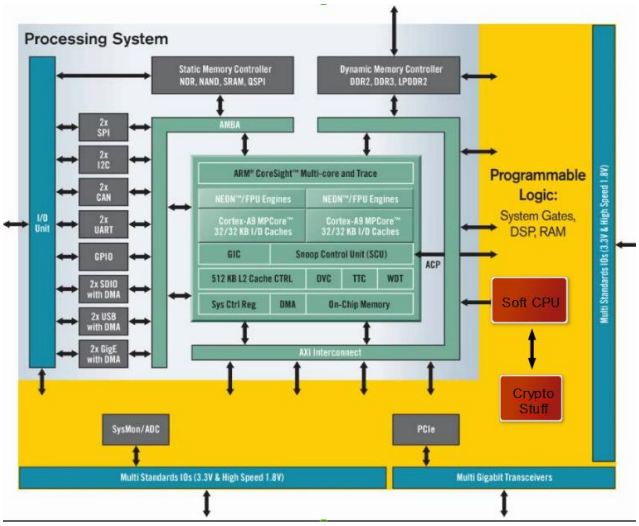
In ALESSIO, **two kinds of updatability** are considered:

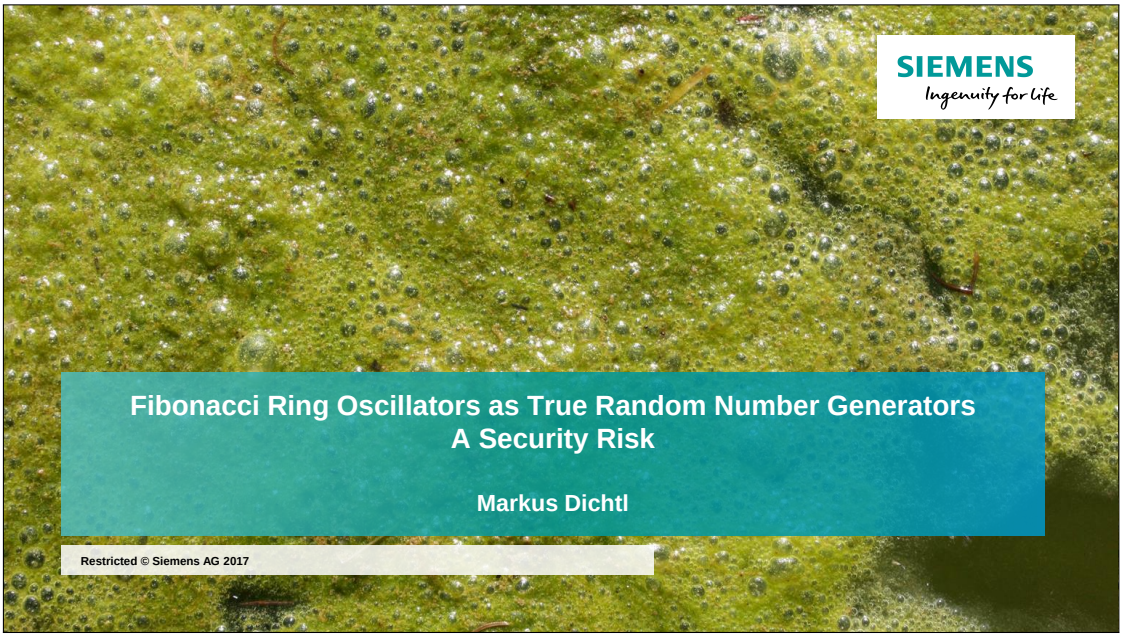
- **Software updates**
- **Updates of FPGA configurations**

Infineon and G&D are only interested in software updates, the rest only in FPGA updates.

The ALESSIO FPGA structure

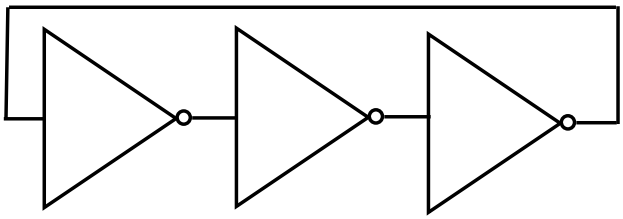
SIEMENS
Ingenuity for life





Why classical ring oscillators take so long to achieve randomness?

SIEMENS
Ingenuity for life



Jitter means random variations in the period length of the RO. But most jitter contributions are (partially) eliminated by others. It takes n periods to achieve an accumulated jitter proportional to √n.

Faster Digital TRNGs

SIEMENS
Ingenuity for life

In 2004, Jovan Golić (Telecom Italia) invented two strongly improved variants of ROs,

Fibonacci ring oscillators (**FIRO**) and

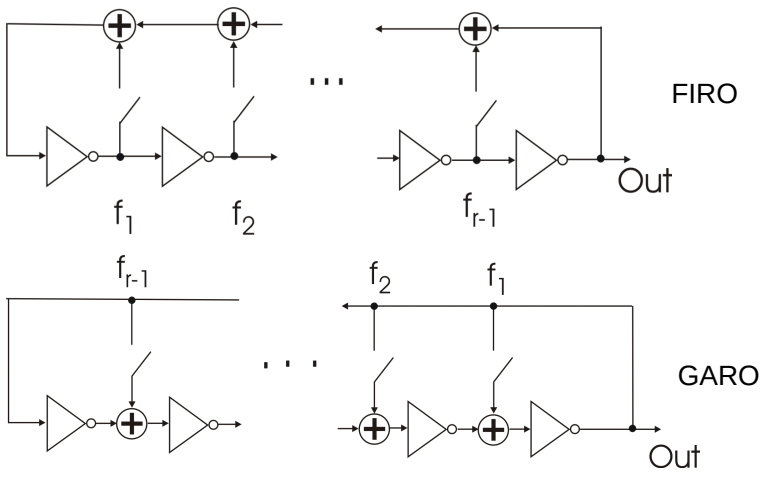
Galois ring oscillators (**GARO**)

The designs show similarities with Fibonacci and Galois LFSRs, albeit the registers are replaced with inverters

J. Dj. Golić , “New Methods for Digital Generation and Postprocessing of Random Data,” IEEE Trans. Computers, vol. 55(10), pp. 1217-1229, Oct. 2006

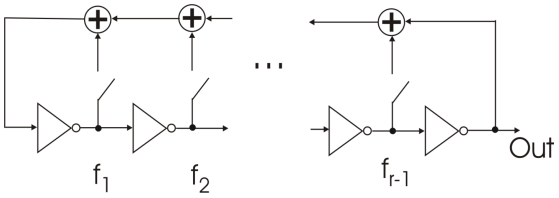
FIROs and GAROs

SIEMENS
Ingenuity for life



How can FIROs and GAROs achieve randomness faster?

SIEMENS
Ingenuity for life

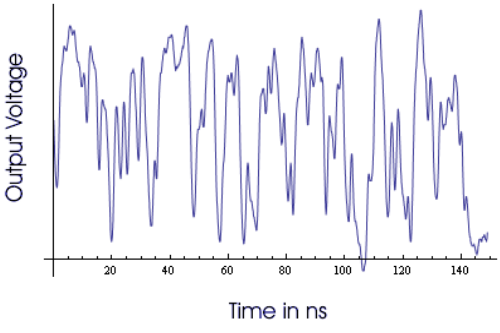


When a signal is subject to a random timing variation, very quickly afterwards the variation branches out to several copies of the signal. This drastically reduces the risk of the jitter contribution to be eliminated.

FIROs and GAROs are assumed to oscillate chaotically.

Starting a FIRO from a static state

SIEMENS
Ingenuity for life

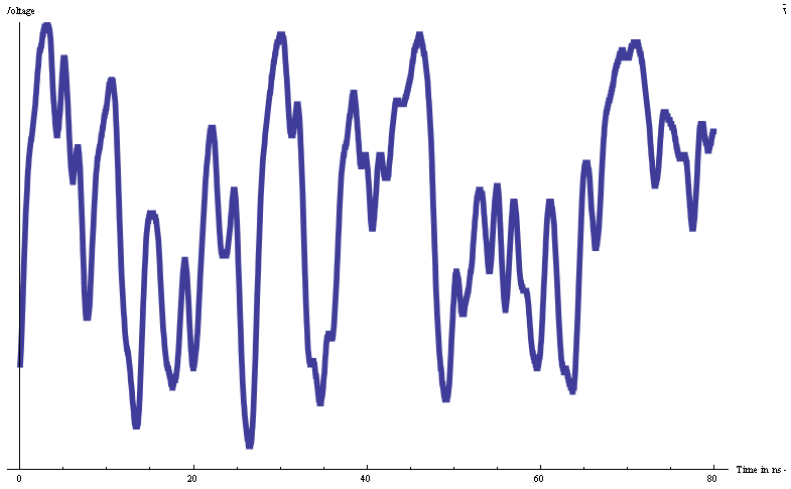


Is this random?

(Feedback polynomial $x^{15}+x^{14}+x^7+x^6+x^5+x^4+x^2+1$)

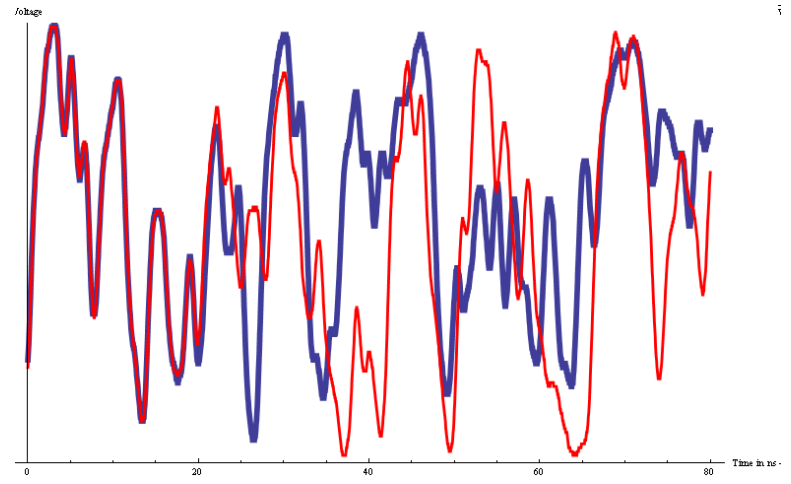
Restarting a FIRO from identical states I

SIEMENS
Ingenuity for life



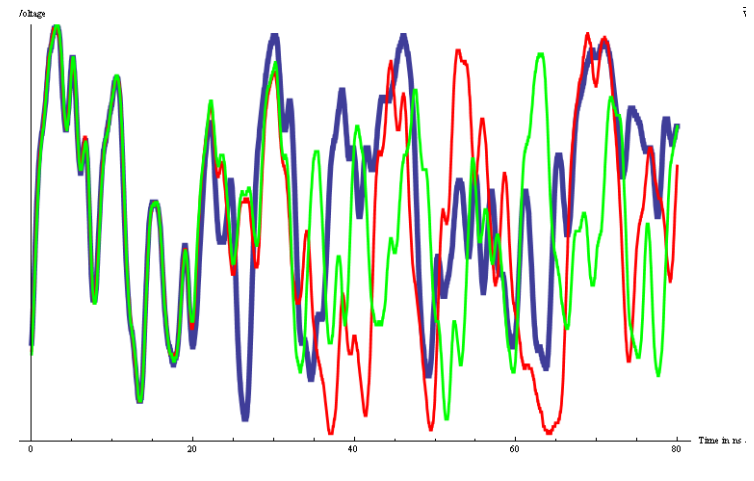
Restarting a FIRO from identical states II

SIEMENS
Ingenuity for life

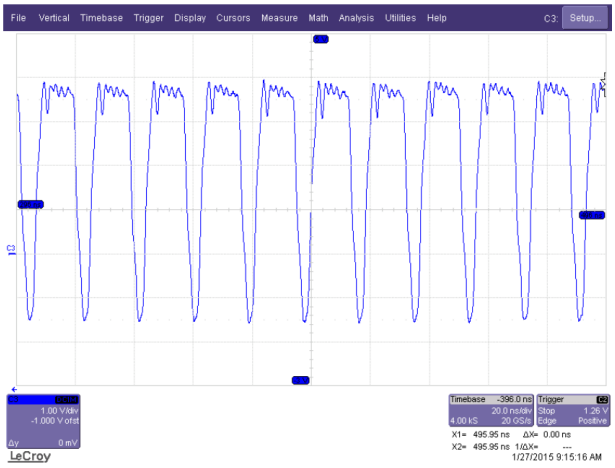


Restarting a FIRO from identical states III

SIEMENS
Ingenuity for life



But...



Oscillogram of a periodic FIRO oscillation

Feedback polynomial
$x^{16} + x^7 + x^2 + 1$

Periodic FIRO oscillations were already reported for very short FIROs in the paper

„Dichtl, Golic: Highspeed True Random Number Generation with Logic Gates Only“ at CHES 2008

For the periodic oscillations, jitter accumulates as poorly as for classical ROs

Many questions

Is this a very rare phenomenon?

How rare?

Is it really periodic?

What does it really mean to be periodic?

How to find out periodicity automatically?

Can we live with TRNGs which **probably** work correctly?

Looking for a suitable quantitative evaluation

Evaluation of thousands of oscilloscope traces seemed doable (automatically), but not really desirable.

The prefered approach was the determination of periodic behaviour on the FPGA board.

The idea was to **sample all inverters of the FIRO simultaneously** and periodically in order to extract as much information as possible about the state.

Temporarily separated: Aquisition of the data and their download from the FPGA board.

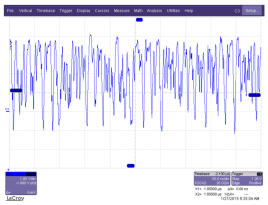
The distinguishing feature

SIEMENS
Ingenuity for life

For **periodic** oscillations, there are **few different patterns sampled**.
For **chaotic** oscillation, there are **many different patterns sampled**.

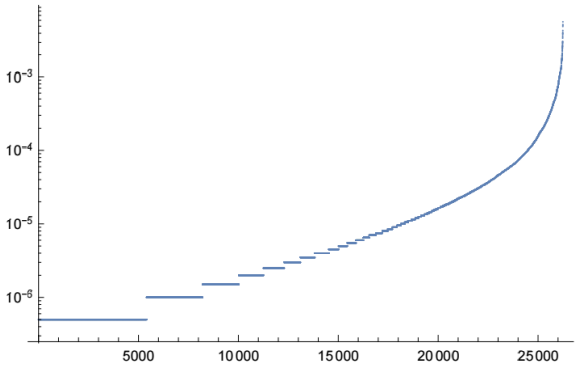
Sampling every 600ns, 10000 samples:
Previous periodic case: only 63 different samples

Chaotic case, feedback polynomial $x^{16} + x^6 + x^3 + 1$: 3442 different samples



For the chaotic case, probabilites of the patterns vary enormously

SIEMENS
Ingenuity for life



Sorted probabilities (logarithmic ordinates) of 2 million samples from a chaotic FIRO oscillation. The feedback polynomial was again $x^{16} + x^6 + x^3 + 1$

Systematically trying (almost) all FIROs of length 16

SIEMENS
Ingenuity for life

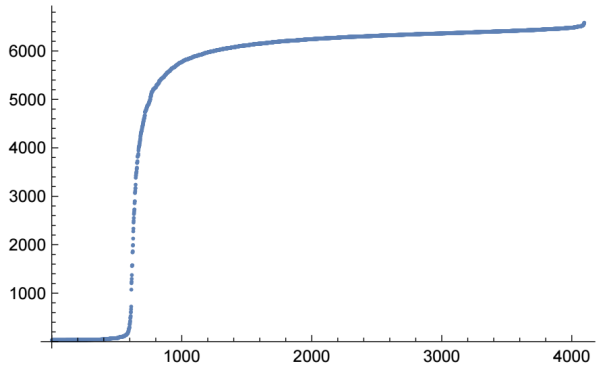
The fixed point free FIROs of length 16 were tried out systematically, however feedback from the first inverter output to ist input was excluded. This resulted in 4096 feedback configurations.The systematic generation was implemented on the FPGA.

In each case, the FPGA was started from a fixed inital state, and 10000 16 bit samples were taken with a temporal separation of 600ns. The samples were stored in RAM.

For each feedback configuration, the number of different samples was determined.

The number of different samples for FIRO length 16, sorted

SIEMENS
Ingenuity for life



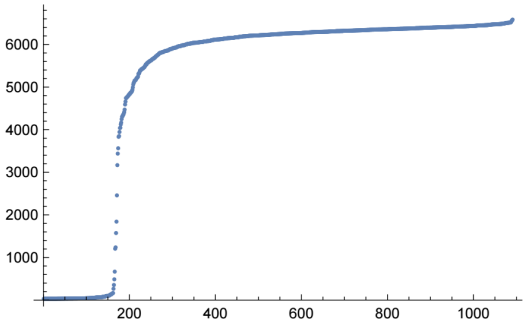
Among the 4096 FIROs of length 16, in 542 cases (13.23%) 100 or less different samples were observed, a clear indication of periodic oscillation.

A TRNG design which fails with a probability of 13.23% inacceptible.

What about the intermediate cases?

Are the failures due to ignoring a irreducibility criterion required by Jovan Golic?

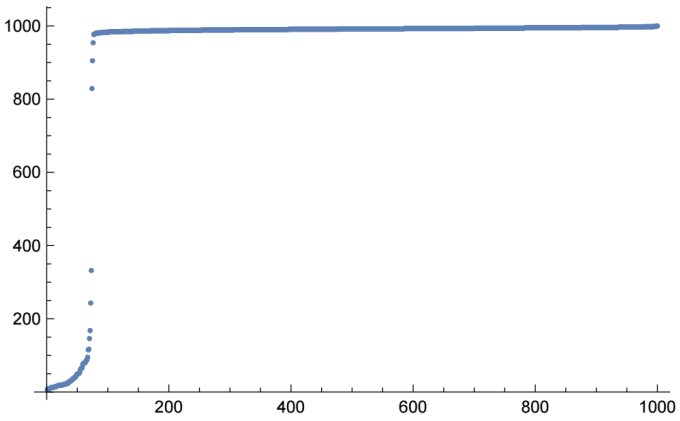
SIEMENS
Ingenuity for life



In order to determine the influence of the Golic irreducibility criterion, the subset of 1091 cases of the 4096 which meet the Golic criterion were evaluated separately.
In 13.84 % of the cases, there were 100 or less different patterns.
This does not imply that the Golic criterion makes things worse. This is just statistics.

Do longer FIROs help?

SIEMENS
Ingenuity for life



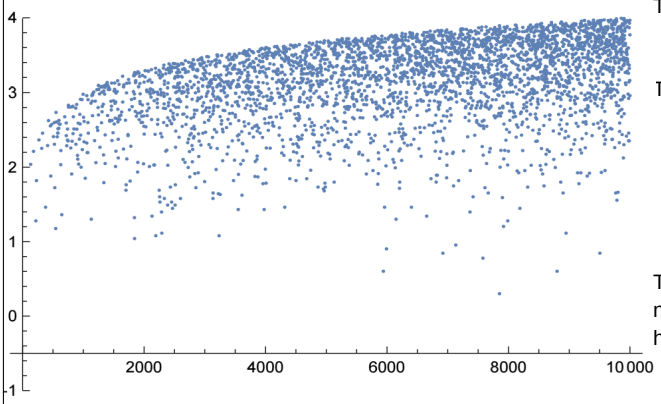
Somewhat. For 1000 random fixed point free FIROs of length 32 meeting the Golic criterion, the number of different 32-bits samples among the 1000 taken was determined.

In 6.7 % of the cases, less than 100 different patterns occurred.

However, it is quite impossible to compare the 16- and 32-bit results.

A new kind of images helps to understand FIROs with intermediate numbers of different samples

SIEMENS
Ingenuity for life



The abscissa represents the sequence of samples in time.

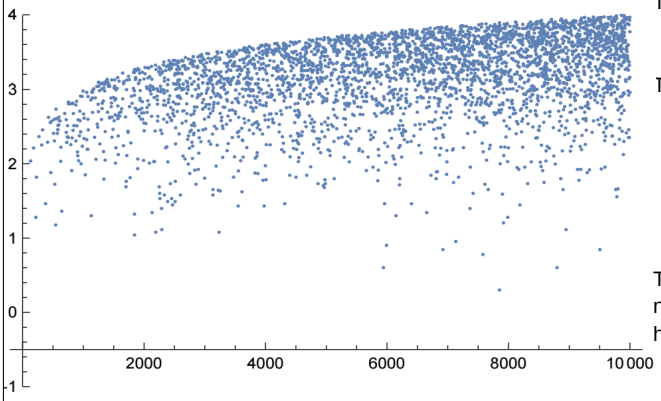
The ordinate of a dot represents how many samples earlier the pattern sampled at the time of the abscissa was sampled the last time previously.If the sample has not occurred previously, there is no dot.

The figure shows the decimal logarithm of the number of samples the sampled 16 bit value has been sampled before for the last time

Permanently chaotic oscillation

A new kind of images helps to understand FIROs with intermediate numbers of different samples

SIEMENS
Ingenuity for life



The abscissa represents the sequence of samples in time.

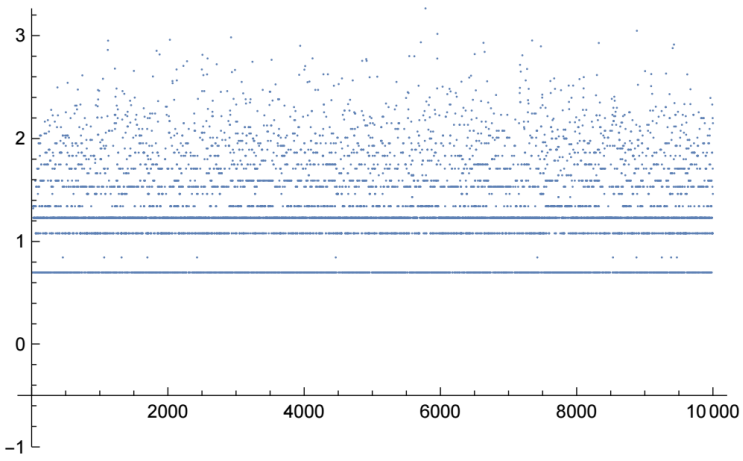
The ordinate of a dot represents how many samples earlier the pattern sampled at the time of the abscissa was sampled the last time previously.If the sample has not occurred previously, there is no dot.

The figure shows the decimal logarithm of the number of samples the sampled 16 bit value has been sampled before for the last time

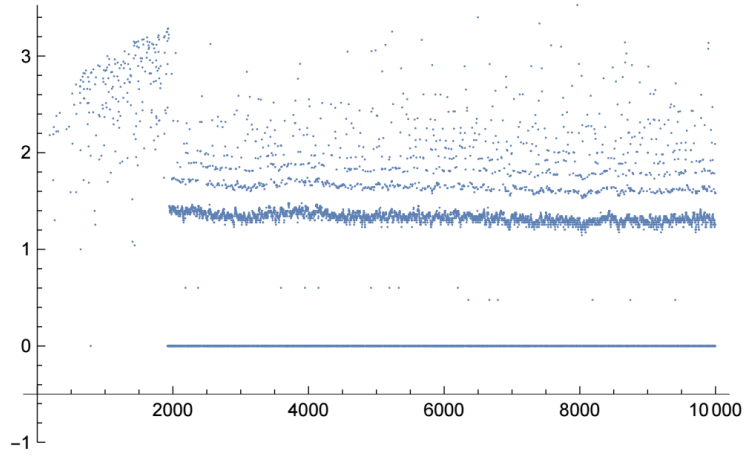
Permanently chaotic oscillation

The new kind of image for a periodic oscillation

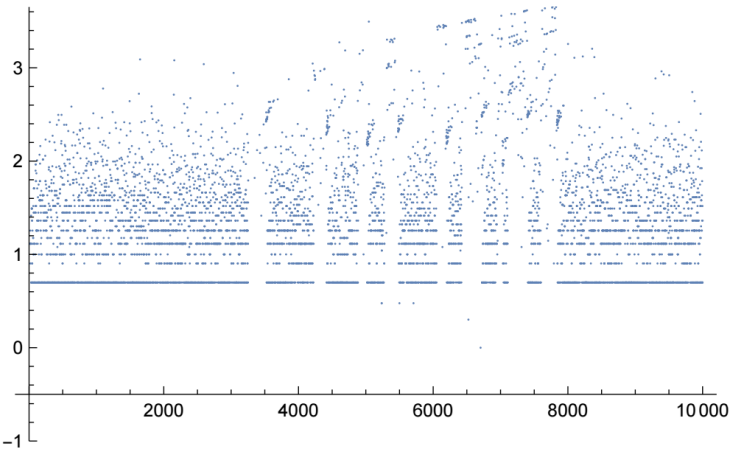
SIEMENS
Ingenuity for life



FIRO starting chaotically, continuing periodically



FIRO starting periodically, occasional chaos



Conclusions

- TRNGs remain a challenging topic
- Periodic FIROs may be detected by the methods used in this presentation, but this requires much more FPGA resources
- There are no good or bad feedback polynomials, as the behaviour depends strongly on the individual chip
- To use GAROs instead of FIROs would be the wrong conclusion, as they also can oscillate periodically

Design and evaluation of a physical random number generator

Guideline for certification targeting high-security applications

Viktor FISCHER

Laboratoire Hubert Curien, UMR 5516 CNRS
Jean Monnet University, Member of University of Lyon
Saint-Etienne, France

fischer@univ-st-etienne.fr

CryptArchi, Smolenice, June 2017

HECTOR

LABORATOIRE HUBERT CURIEN

Motivation and Objectives

- French DGA (Direction Générale de l'Armement) is responsible for security in high-security cryptographic applications.
- French RNG evaluation scheme is based on the German document AIS 20/31.
- DGA considered that for high-end security applications some additional guideline is necessary to complete AIS 31 (the PTRNG part).
- In 2017, David Lubicz edited the document: "**Design and evaluation of a physical random number generator integrated in an electronic chip**"
- **Our objectives**
  - Present briefly the document (Viktor)
  - Illustrate the DGA document on a PLL TRNG design (Elie)

LABORATOIRE HUBERT CURIEN

Outline

LABORATOIRE HUBERT CURIEN

Outline

LABORATOIRE HUBERT CURIEN

Introduction

- The document describes and defines the essential elements of an approach to TRNG design that ensures its security and takes the most recent advances into account.
- The aim of presented approach is to attain the highest level of certainty for the quality of the randomness produced by a generator using an upper limit on the statistical bias that can be measured at the generator output.
- The approach is described in a series of requirements that are well-founded and argued.
- Some definitions that appear important and which are perhaps insufficiently clarified in the literature are also provided.

LABORATOIRE HUBERT CURIEN

Outline

LABORATOIRE HUBERT CURIEN

Introduction

- TRNG – physical device producing a series of unpredictable bits.
- The operation of the TRNG must rely on a random physical phenomenon known as analog physical noise and must include an analog-to-digital converter (ADC).
- We will first consider the analog physical noise as a source of randomness.

LABORATOIRE HUBERT CURIEN

Definition 1 – Internal stage of the generator

- Random number generator – a physical device G of internal state $E : t \to V$ depending on time t, with value in a space of phases V and producing a series of bits $b_1(t_1)b_2(t_2)\ldots$.
- The value of an output bit at a given time knowing the internal state is perfectly determined.
- Comments:
  - This means that the change in the internal state takes into account the full random nature of the generator.

LABORATOIRE HUBERT CURIEN

Requirement 1 – Identification of the source of randomness

- **The phenomenon of physical noise responsible for the unpredictable nature of generator operation must be clearly identified.**
- Comments:
  - Unidentified phenomena may contribute to the random nature of the operation of the TRNG, too.
  - They shouldn't be taken into account in entropy estimation.

LABORATOIRE HUBERT CURIEN

Definition 2 – Statistical model of the physical noise

- Statistical model of the physical noise – a stochastic model of time variable t with value in the space of phases V describing the change of $E(t)$.
- It may appear as a probability distribution $\mathbb{P}(E(t)|p_1,\ldots,p_n,E(t_0)=\ldots)$, with $t > t_0$ on $E(t)$, depending on parameters $p_1,\ldots,p_n$ and preconditions on $E(t_0)$.
- We make the assumption that the distribution $\mathbb{P}(E(t)|p_1,\ldots,p_n,E(t_0))$ contains all the information accessible to an observer (whatever his calculation power) with knowledge of the preconditions on $E(t_0)$.
- Afterwards, such a statistical model is denoted $M(t,p_1,\ldots,p_n)$.

LABORATOIRE HUBERT CURIEN

Requirement 2 – Characterization of the physical noise

- **There must be a statistical model $M(t,p_1,\ldots,p_n)$ for the physical noise used.**
- Comments:
  - The parameters (e.g. temperature, supply voltage) and the preconditions (e.g. initial phase) are assumed to be known to the attacker.
  - They can be manipulated by the attacker but only within certain limits.
- Remarks:
  - The model can only be deduced from an explanation and physical modelling of phenomena.
  - A statistical analysis of the physical noise, e.g., using statistical tests is insufficient.

LABORATOIRE HUBERT CURIEN

Requirement 3 – Experimental evaluation of input parameters of the noise model

- **One must be able to evaluate experimentally the parameters $p_1,\ldots,p_n$ of the statistical model for physical noise $M(t,p_1,\ldots,p_n)$.**
- **One must be able to evaluate the measurement errors of these parameters.**
- Comments:
  - Parameters can be evaluated externally or internally.
  - External measurements can use high-end measurement tools, but:
    - Measurement can be unprecise, because of data interface.
    - It can be difficult to implement on a production line and complicate testing each device.
  - Here, the use of statistical tests is legitimate.

LABORATOIRE HUBERT CURIEN

Requirement 4 – Evaluation of stability of noise model parameters in time

- **The stability of the parameters $p_1, \ldots, p_n$ of the statistical model must be evaluated for physical noise with regard to:**
  - **physical environmental operating conditions of the RNG: temperature, supply voltage, electromagnetic environment;**
  - **technological environmental operating conditions of the RNG: installed alone on a circuit or with other circuits (e.g., enryption);**
  - **different integrations of the generator (depending on the target technology).**

  Aging tests could also be performed.

- Comments:
  - Requirements 3 and 4 can be called the technology qualification.
  - To perform it, circuits should be designed to ensure that:
    - measurements are as accurate as possible;
    - the circuit is tested in the most unfavourable environmental conditions possible.

LABORATOIRE HUBERT CURIEN

Outline

LABORATOIRE HUBERT CURIEN

Introduction

- An analog-to-digital converter produces a series of bits that is a deterministic function of the internal state of the generator.

LABORATOIRE HUBERT CURIEN

Definition 3 – Statistical model of the complete TRNG

- A statistical model for the TRNG whose model of physical noise is $M(t, p_1, \ldots, p_n)$ is a stochastic model $N(t, p_1, \ldots, p_n, q_1, \ldots, q_m)$ with value in the series of bits with arbitrary length, where $p_1, \ldots, p_n$ are the parameters of the physical noise model and $q_1, \ldots, q_m$ are the parameters of the TRNG.
- Certain parameters $q_1, \ldots, q_m$ must be adjusted during component design (but cannot be manipulated by an attacker).

LABORATOIRE HUBERT CURIEN

Requirement 5 – Availability of the statistical model of the complete TRNG

- **There must be a statistical model for the TRNG.**
  - **This assumes all the conditions given above, including that there must be a statistical model for the physical noise.**
- Comments:
  - This assumes that all the previous requirements are fulfilled.
  - Validation of the model is not simple, especially when the physical noises are known only partially.
  - It is very difficult to take into account global deterministic noises.

LABORATOIRE HUBERT CURIEN

Requirement 6 – Setup of the RNG design parameters

- **Using the statistical model of the TRNG, it must be possible to adjust parameters $q_1, \ldots, q_n$ to limit the bias on the generator output bits with a defined value.**
- Comments:
  - The use of common statistical tests is once again entirely legitimate.

LABORATOIRE HUBERT CURIEN

Definition 4 – Selection of model parameters

- A parametric test for a TRNG of statistical models $N(t, p_1, \ldots, p_n, q_1, \ldots, q_n)$ is a test that verifies that parameters $p_1, \ldots, p_n, q_1, \ldots, q_n$ are in a certain domain that ensures a sufficient entropy rate. The tests are instantaneous if they can operate at the same time as the RNG.
- With this definition, the following requirement can be set

Requirement 7 – Parametric statistical tests and their execution

- **Parametric tests must run at startup and continuously.**
- Comments:
  - A common statistical test cannot be interpreted like a parametric test
  - In this case, it is useless and even dangerous.

Definition 5 – Deterministic test

- A deterministic test for a TRNG is any test that verifies the integrity of the sampling device that associates deterministically an output bit with the value of the internal state.
- With this definition, we can write the next requirement

Requirement 8 – Availability of a deterministic test

- **There must be tests of deterministic functions that verify proper operation of the functional elements of the TRNG.**
- Comments:
  - Proper operation of the digitizer must be included in this test.

Outline

Conclusions

- Requirements presented in this document represent an extension of those given in AIS 31 (requirements of AIS 31 remain valid).
- Because the highest security levels are targeted by the document, comparing to AIS 31, some additional requirements are given:
  - Statistical model of the source of randomness must be given.
  - Deterministic part of the whole TRNG (not only of the post-processing) must be tested.
  - Parametric tests must be based on the statistical model of the TRNG.
  - General purpose statistical tests should not be used as parametric tests.

Acknowledgments

This work was performed in the framework of the project

HECTOR
Hardware Enabled Crypto and Randomness

The HECTOR project has received funding from the
European Union's Horizon 2020 research and innovation programme
under grant agreement number 644052 starting from March 2015

www.hector-project.eu

An illustration of a new certification approach for True Random Number Generators (TRNG)

Elie Noumon Allini • Florent Bernard • Viktor Fischer

Laboratoire Hubert Curien, UMR 5516 CNRS
Université Jean Monnet, Membre de l'Université de Lyon
Saint-Etienne, France

CryptArchi, Smolenice, June 2017

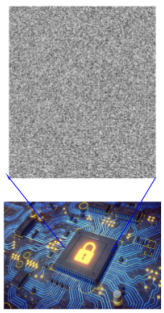
Importance of Random number generators 2

Crucial component of cryptographic systems

- Typical use
  - Key generation,
  - Initialization vector,
  - Counter measures against side channel attacks.
- Security relevance
  - Security of the whole system is based on the secret key
    - ↪ Key must be generated as often as needed,
    - ↪ Unpredictable and non reproducible way;
  - Need of *good* random numbers.

Certification 3

Our device produces very good randomness



Can you prove it?

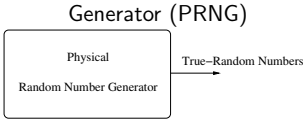
Bundesamt für Sicherheit in der Informationstechnik

Governmental organization

Is the TRNG embedded in your cryptographic device safe enough to ensure a high security level?

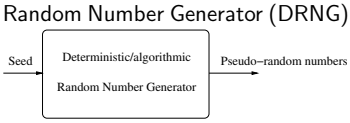
Types of Random Number Generation (RNG): 4
2 classes

Physical Random Number Generator (PRNG)



- Exploits noisy analog phenomenon: thermal noise, flicker noise, ...
- May produce random numbers looking not really random.
- Design challenging.

Deterministic Random Number Generator (DRNG)

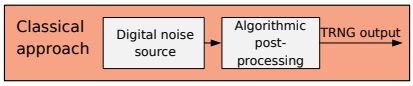


- Take a Seed as input.
- Algorithmic process is known.
- Security based on the Seed
- "Looks like" random numbers.

Question

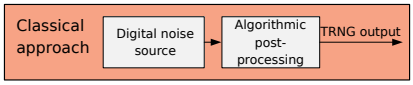
How to evaluate properly the quality of a Random Number Generator?

A first approach for TRNG evaluation: 5
Observation

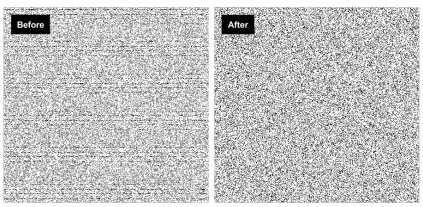


- Battery of statistical tests (FIPS, NIST, DieHard) at the TRNG output.
- Problem 1 : even a full deterministic sequence can pass these tests ⇒ tests are necessary but not sufficient.

A first approach for TRNG evaluation: 5
Observation

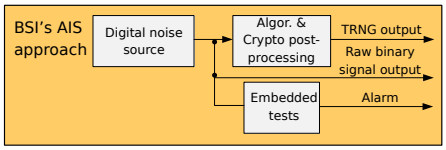


- Battery of statistical tests (FIPS, NIST, DieHard) at the TRNG output.
- Problem 2 :



- Need to perform tests **before** post-processing.

A second approach for TRNG evaluation: proof/certification



Same as the classical approach plus:

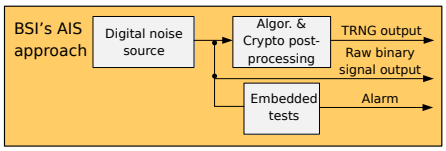
- \+ Test the raw binary signal and estimate the entropy (min entropy) per generated bit
- \+ Provide embedded tests to detect a total failure of the noise source.

Problem : Entropy is not a property of the generated sequence but of the underlying random variables.

Stochastic model

$\Rightarrow$ Need a stochastic model to compute a lower bound (H_{min}) of the entropy per bit as close as possible to the source of entropy.

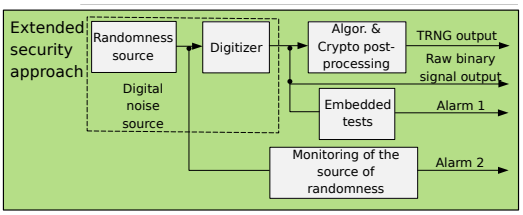
Stochastic Model



- Problems:
  - Model $\neq$ Reality
  - Need reasonable assumptions to work on random variables.
    - ↪ Is the noise composed only of thermal noise[1] as it is almost always assumed in the TRNG state of the art?
  - There is no generic Model: each TRNG principle must be described with a dedicated stochastic model.

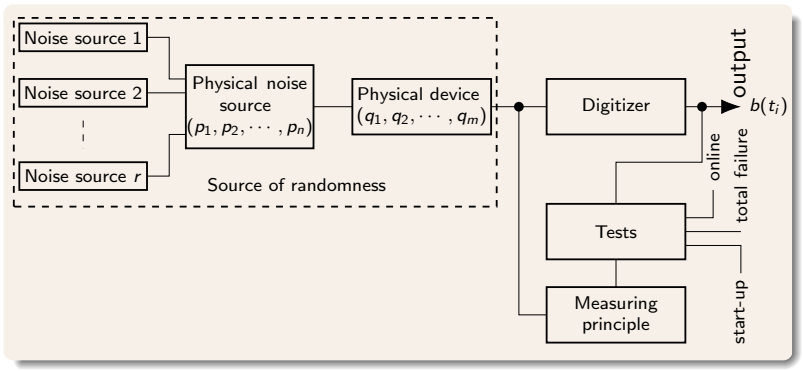
[1] Thermal noise is considered to be unavoidable and non manipulable

3rd approach: Even closer to the entropy source



- Identify and Model each noise source.
- Design digitization process (entropy extractor) to extract maximum entropy.
- Provide a mathematical proof to compute a lower bound of entropy rate.
- Develop **specific** tests of the source of randomness to avoid a total failure of the entropy source and to monitor it "online".
- Work in progress: Workgroup "alea" with DGA-MI, ANSSI, Institut Fourier, LaHC-SESAM

General overview of a hardware-based TRNG 9



Random number generator 11

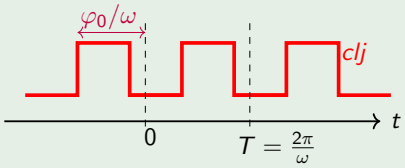
Internal state in the diagram

Formal definition

- Physical device
  - internal state $E : \begin{array}{ccc} \mathbb{R}_+ & \longrightarrow & V \\ t & \longmapsto & E(t) \end{array}$
  - produces a sequence of bits $(b_{t_i})_{1 \leqslant i \leqslant n}$ in some given time.
- Value of b_{t_i} knowing E is determined.

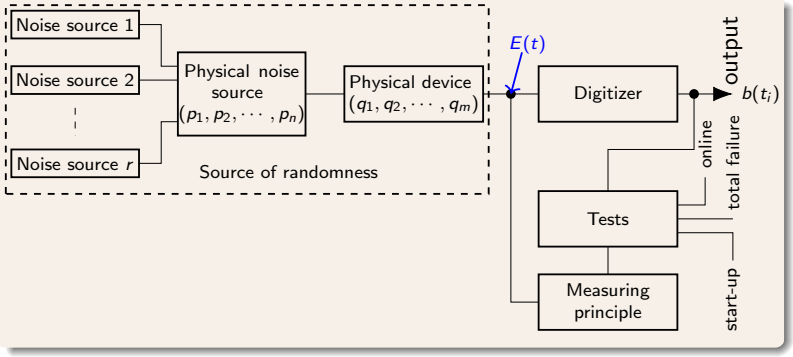
Example

- $clj : t \longmapsto P(\omega t + \varphi_0)$
  - P is 2π-periodic
- $V = [0, 2\pi[$
- $E : t \longmapsto \omega t + \varphi_0 \mod 2\pi$

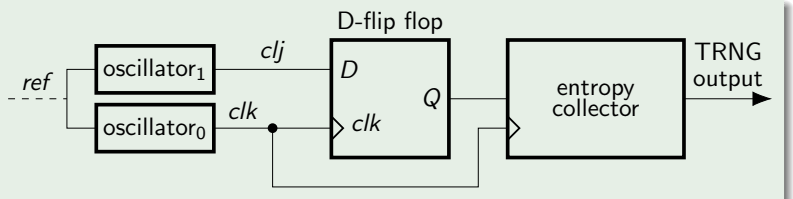


Random number generator 11

Internal state in the diagram



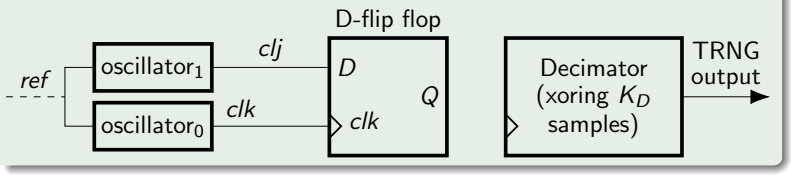
General principle of an oscillator based TRNG 12

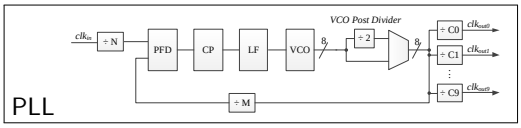


- clk
  - reference signal,
  - *constant period* T_{clk}.
- clj
  - jittery signal;
  - random variable
    - ↪ mean value T_{clj}.

General principle of an oscillator based TRNG 13

PLL-based TRNG

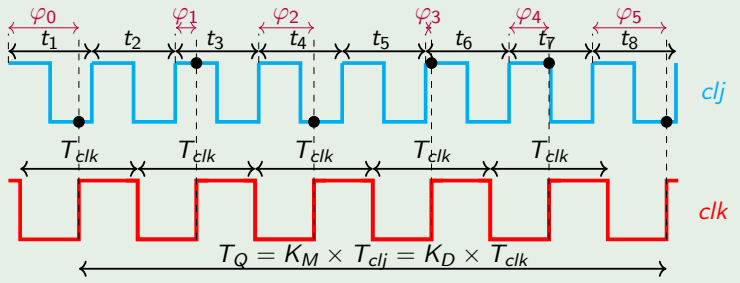




$$\begin{cases} K_M = M \\ K_D = N \times C_0 \\ \frac{F_{clk_{in}}}{F_{clk_{out0}}} = \frac{K_M}{K_D} \end{cases}$$

♣ Control of the phase difference between input and output signal of the PLL ⇒ Control of the drift that remains bounded.

Sampling example 14

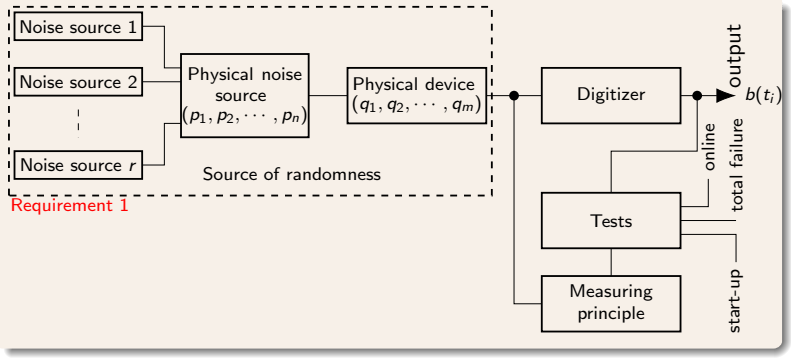


- t_i : realizations of T_j;
- φ_i : phase of clj at time iT_{clk},
  - ↪ $E(iT_{clk})$
  - ↪ $iT_{clk} + \varphi_0 \mod T_{clj}$
- K_M : number of cycles of clj;
- K_D : number of samples,
  - ↪ number of cycles of clk.

Identification of the noise source 15

Requirement 1

The physical phenomenon at the origin of the unpredictable character of the generator operation must be well identified.



Identification of the noise source 15

Requirement 1

The physical phenomenon at the origin of the unpredictable character of the generator operation must be well identified.

Sources of entropy in PLL-TRNG

- Differential jitter (dynamic difference in phases)
  - between clock signals generated in two PLLs connected in parallel.
- Three sources can be recognized:
  - input clock jitter,
  - intrinsic noise of the PLL,
  - supply noise contributing to the PLL output clock jitter.

Input clock jitter 16

- Input jitter with frequency lower than the PLL bandwidth :
  - passed by the PLL without being modified (not filtered out).
- When frequency corresponds to the PLL bandwidth :
  - the closed loop transfer function of the PLL features a peak,
  - input jitter amplified by the relative size of the peak,
    ↪ depending on the loop damping factor.
- When frequency is higher than the PLL bandwidth
  - input jitter is attenuated at 20db/decade.

Conclusions

- Jitter of the input clock should be as small as possible
  - limit the PLL output jitter to the PLL intrinsic jitter,
  - use of quartz.
- Input clock frequency should be as high as possible
  - much higher then the PLL bandwidth.

Intrinsic noise of the PLL 17

- VCO contributes the most to PLL intrinsic noise.
- Main components of the PLL intrinsic noise :
  - Thermal noise
  - Flicker noise
- Flicker noise may be significantly reduced :
  - appropriate selection of multiplication and division factors.

Conclusions

- Output clock frequency should be as high as possible
  - reduce the contribution of the flicker noise.
- PLL bandwidth should be as large as possible
  - reduce the long term jitter at PLL output.

Supply noise contribution 18

- Analog and digital supply noises contribute to the jitter :
  - any fast (step) variation on the analog supply of the PLL
    - ↪ instantaneous change in VCO frequency,
    - ↪ reflected as jitter at the PLL output clock.
  - any step variations on the digital supply of the PLL
    - ↪ a variation in the delay of the digital circuits,
    - ↪ result in variation of the clock time period,
    - ↪ reflected as time period jitter at the PLL output.
- Time period deviation is independent of the PLL output time period
  - unlike the deviation due to intrinsic device noises.

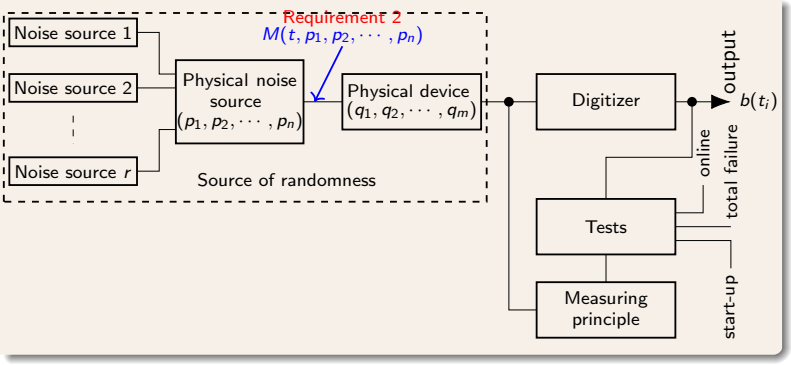
Conclusions

- Analog and digital power supplies should use
  - linear regulators and a high quality filters;
- Digital power supplies powering the PLL
  - must not power also the FPGA core.

Characterization of the noise 19
Stochastic model

Requirement 2

The physical noise must have a stochastic model $M(t, p_1, p_2, \cdots, p_n)$.



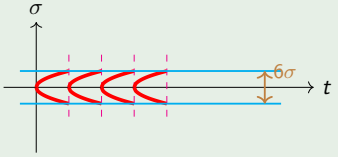
Characterization of the noise 19
Stochastic model

Requirement 2

The physical noise must have a stochastic model $M(t, p_1, p_2, \cdots, p_n)$.

In the case of a PLL-based TRNG

- $M(t, \sigma) \equiv \mathcal{N}(0, \sigma)$
  - $\sigma = \sqrt{K_M} \times \sigma_{VCO}$.



Requirement 2bis

The physical noise stochastic model $M(t, p_1, p_2, \cdots, p_n)$ should be used to get a probability distribution of TRNG the internal state $(\varphi(t))$.

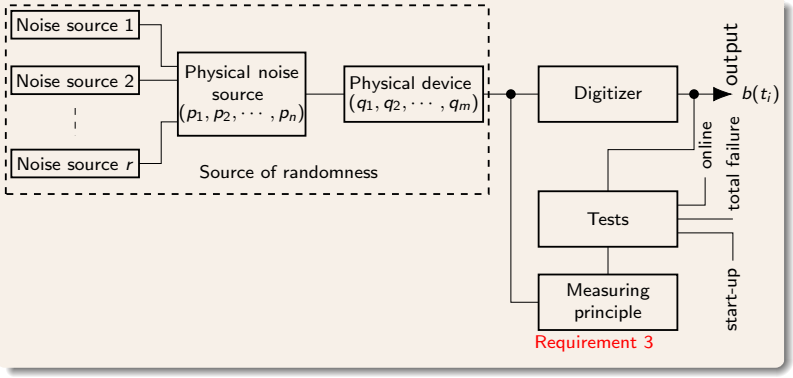
$$\mathbb{P}(\varphi(t)|p_1, p_2, \cdots, p_n, \varphi(t_0)).$$

Parameters evaluation 20

Requirement 3

- It should be possible to evaluate the initial state φ_0.
- It should be possible to experimentally evaluate $p_1, p_2, \cdots, p_n$.
- It should be possible to evaluate measurement errors.

Parameters evaluation 20



Parameters evaluation 20

Requirement 3

- It should be possible to evaluate the initial state φ_0.
- It should be possible to experimentally evaluate $p_1, p_2, \cdots, p_n$.
- It should be possible to evaluate measurement errors.

Measurement techniques

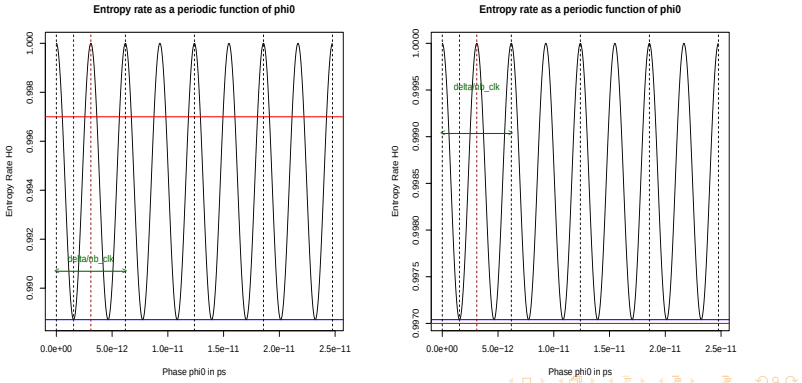
- φ_0 can be quite difficult to evaluate.
- The number of unstable samples is directly related to parameter $p_1 = \sigma$.
- The duty cycle can be approximated by $\frac{\#\{X_i=1\}}{K_D}$.

A way to circumvent evaluation of φ_0

A conservative approach

Worst case

Set a higher σ_{min} such that for **any** φ_0 we have $H_{min} \geqslant 0.997$.



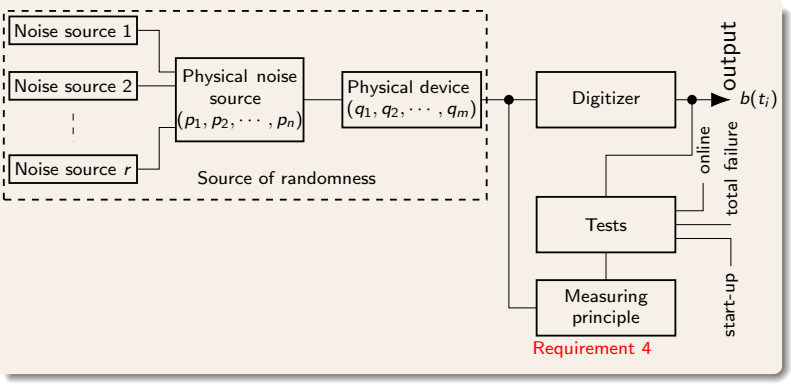
Stability of parameters

Requirement 4

Stability of parameters $p_1, \cdots, p_n$ should be evaluated with regard to :

- physical environmental conditions (tmperature, voltage, etc),
- technological environmental conditions,
- aging tests.

Stability of parameters



Stability of parameters 22

Requirement 4

Stability of parameters $p_1, \cdots, p_n$ should be evaluated with regard to :

- physical environmental conditions (tmperature, voltage, etc),
- technological environmental conditions,
- aging tests.

Case of the PLL-based generator

- Jitter (σ) is bounded and duty cycle (α) close to 0.5 at the output,
  - model should tolerate slight unbalances.
- Tests still on progress.

Statistical model of a TRNG 24

Definition

- Statistical model of the TRNG
  - stochastic model $N(t, p_1, p_2, \cdots, p_n, q_1, q_2, \cdots, q_m)$
    - $\hookrightarrow$ $p_1, p_2, \cdots, p_n$ parameters of the physical noise model,
    - $\hookrightarrow$ $q_1, q_2, \cdots, q_m$ parameters of the TRNG;
  - values in the set of sequences of bits of arbitrary length.
- About parameters $q_1, q_2, \cdots, q_m$
  - some should be adjustable,
  - none should be manipulable (by an attacker).

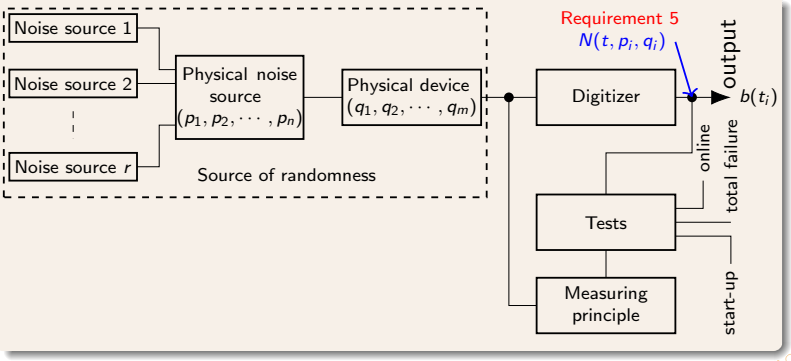
Parameters in the case of a PLL-based TRNG

- Initial internal state : φ_0.
- σ, α.
- K_M, K_D.

Statistical model of a TRNG 25

Requirement 5

A statistical model of the TRNG should be available and should use the probability distribution $\mathbb{P}\big(\varphi(t)|p_1, p_2, \cdots, p_n, \varphi(t_0)\big)$.



Statistical model of a TRNG 25

Requirement 5

A statistical model of the TRNG should be available and should use the probability distribution $\mathbb{P}\big(\varphi(t)|p_1, p_2, \cdots, p_n, \varphi(t_0)\big)$.

In the case of a PLL-based TRNG

- X_i random variable with values in $\{0, 1\}$
  - logical level of the sampled bit at time iT_{clk}.
- Probability to sample bit 1 at $i \times T_{clk}$[a]

$$\mathbb{P}(X_i = 1) \;=\; \mathbb{P}(\varphi_i < \alpha T_{clj}) - \mathbb{P}(\varphi_i < 0) + 1 - \mathbb{P}(\varphi_i < T_{clj}).$$

[a] F. Bernard, V. Fischer, B. Valtchanov. Mathematical Model of Physical RNGs Based On Coherent Sampling. Tatra Mountains - Mathematical Publications, 2010.

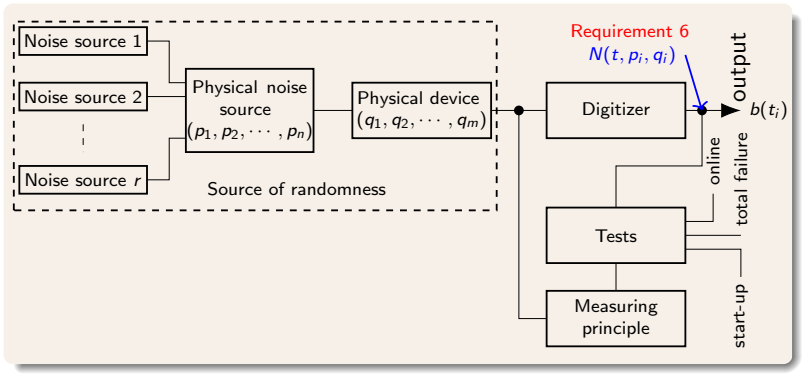
Use of the statistical model of a TRNG 26

Obtaining the best configuration

Requirement 6

From the statistical model of the TRNG, it should be possible to adjust parameters $q_1, q_2, \cdots, q_m$ in order to bound the value defined by the bias on the bits that output from the generator.

Use of the statistical model of a TRNG
Obtaining the best configuration



Evaluation process Conclusion Physical noise Analog to digital converter

Use of the statistical model of a TRNG
Obtaining the best configuration

26

Requirement 6

From the statistical model of the TRNG, it should be possible to adjust parameters $q_1, q_2, \cdots, q_m$ in order to bound the value defined by the bias on the bits that output from the generator.

In the case of a PLL-based TRNG

- Combinatorial optimization
  - heuristic and metaheuristic methods.
- Work still in progress.

E. Noumon Allini • F. Bernard • V. Fischer An illustration of a new certification approach for TRNG

Evaluation process Conclusion Physical noise Analog to digital converter

Use of the statistical model of a TRNG
Monitoring the source of entropy

Requirement 7

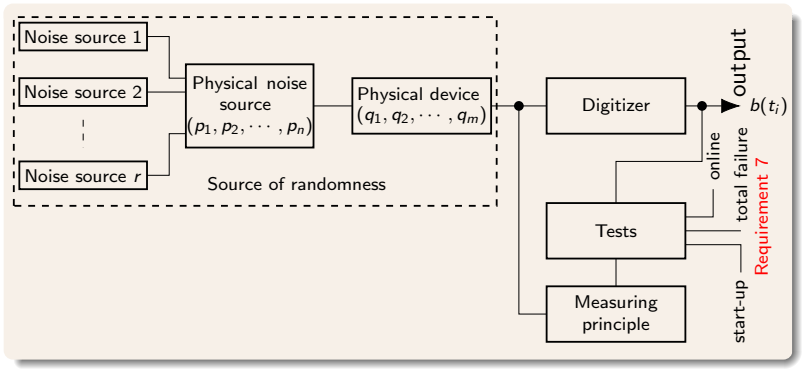
Some parametric tests should be available

- at start-up,
- online,
- total failure test.

E. Noumon Allini • F. Bernard • V. Fischer An illustration of a new certification approach for TRNG

Use of the statistical model of a TRNG 27

Monitoring the source of entropy



Use of the statistical model of a TRNG 27

Monitoring the source of entropy

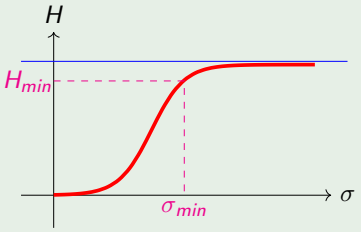
Requirement 7

Some parametric tests should be available

- at start-up,
- online,
- total failure test.

A parametric test

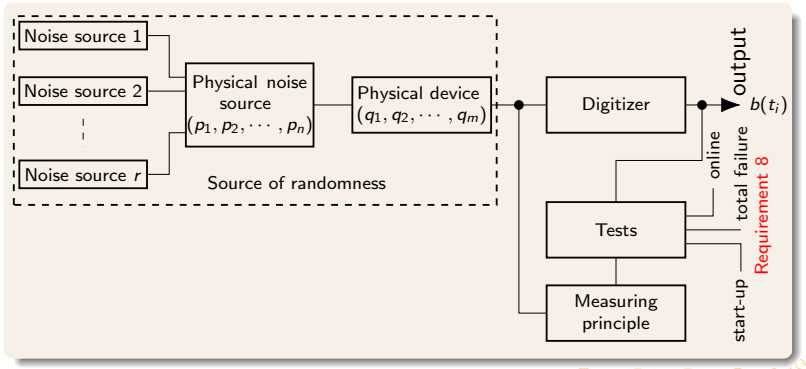
- $H = f(\sigma | K_M, K_D, \varphi_0, \alpha)$
- Entropy threshold H_{min}
  - minimum entropy tolerated
    - ↪ alarm (under H_{min});
  - related to σ_{min}.
- Online test.



Deterministic tests 28

Requirement 8

There must be tests of deterministic functions that verify proper operation of the functional elements of the TRNG.



Feasibility and pertinence of the approach 30

Higher security level

- Take into account the source of randomness
  - not only the output or digitized noise.

Illustration of the approach

- Evaluated on the PLL-based TRNG
  - approach valid.
- Evaluated on the RO-based TRNG
  - approach valid.

General process?

- Other hardware generators
  - TERO,
  - STR,
  - ...

Attack tree construction: an application to the connected vehicle

CryptArchi 2017
June 18-21 Smolenice - Slovakia

Khaled karray, Jean-luc Danger, Sylvain Guilley, Moulay Abdelaziz El Aabid

Presentation Outline

Introduction
- Cyber-physical architecture
- Attacks on cars
- Protections and Countermeasures

Risk assessment and attack tree generation
- Problem statement
- Approach to attack modeling

Conclusion

Plan

Introduction
- Cyber-physical architecture
- Attacks on cars
- Protections and Countermeasures

Risk assessment and attack tree generation
- Problem statement
- Approach to attack modeling

Conclusion

Vehicle architectural components (1)

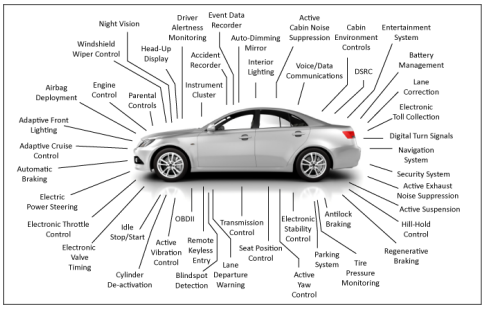


Figure : Automotive services

Cost:

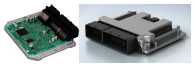
The cost of electronics embedded system in the vehicle is estimated to be between 20% and 25% of total production cost

Vehicle architectural components (2)

- Sensors: Used to collect information, data measurements of the car environment (speed, radar, temperature . . .)

- Actuator: Elements of the vehicle architecture that converts “commands” into “ actions” (exp: engine, wheel orientation . . .)

- ECUs: Electronic Control Units, are embedded interconnected controllers in the vehicle and integrate various data processing functions that guarantee the safety and comfort of the user.



Vehicle architectural components (3)

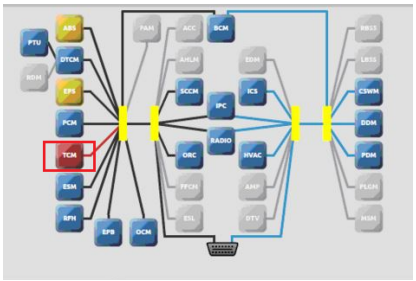
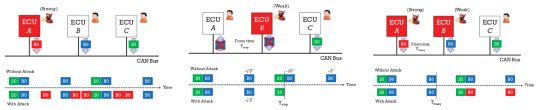


Figure : Vehicle architecture

A vehicle architecture consists of these components interconnected by communication buses: CAN, FlexRay, . . .

Attacks on cars (1)

- Attack with physical access to the vehicle:
  - CAN frame injection attack [Impersonation].



(a) Fabrication attack. (b) Suspension attack. (c) Masquerade attack.

  - DoS-attack (Denial of Service)

While (True) {Send CAN ID=0, payload = 0 }

  - Attacks exploiting software vulnerabilities to access the internal networks (e.g CAN).

Attacks on cars (2)

- Attacks with short/long range wireless access:
  - Wifi, Bluetooth, Cellular

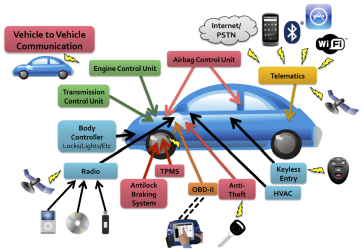
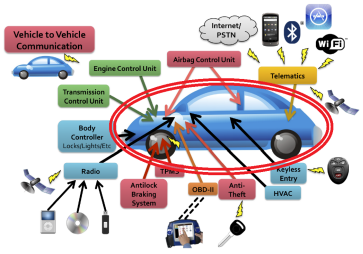


Figure : Vehicle attack vectors

Protections and Countermeasures: Vehicle isolation

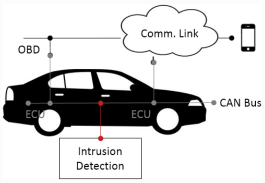


- Firewalls [Wifi, Bluetooth, Cellular . . .].
- Secure connections with off-board equipment.
- Importing and adapting known solutions from IT world to the automotive world.
- . . .

What happens if these protection layers are bypassed ?

Protections and Countermeasures: Intrusion Detection and Internal Bus Protection

Intrusion detection/prevention on CAN Bus



IDS/IPS solutions begin to emerge in the market.

Security primitives on internal Buses

- Using security primitives in internal communications buses
  - Encryption of data on CAN network.
  - Authentication of source and destination on CAN network.
  - ...

Plan

Problem statement:

- Car manufacturers have to guarantee a level of security: safety, privacy of the user.
- Some design problems have to be addressed at early design phase.
- Write clear and strong security requirements to meet the security needs.

Problem statement:

1. Asset identification.
2. Threat identification (for each asset).
3. Impact estimation (for each threat).
4. Threat level estimation [likelihood] (for each threat) :
   - (1) architectural design
   - (2) Adversary capabilities

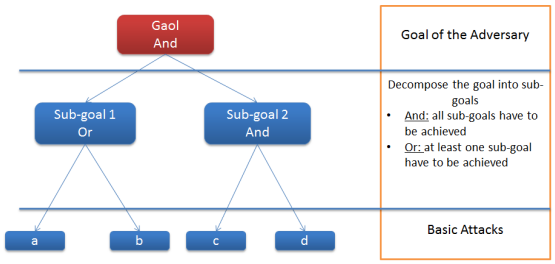
How to assess the risk

$$SC = \{SC_i\}$$

$$Risque = \sum_i Impact(SC_i) \times P_{occ}(SC_i)$$

SC	:	Set of attack scenarios
$Impact(x)$	:	Impact of the attack scenario on safety, availability, confidentiality, ...
P_{occ}	:	Likelihood of occurrence of the attack scenario is correlated with the architecture (functional, hardware)

Attack tree



Goal of the attack tree:

- Visualise attack paths (understand *How*
- Helps to assess the Risk.
- Allows to think about countermeasure in an effective way.

This approach is used to support the risk assessment procedure.

Using attack trees

Attack trees are used to model attacks:

- Network administration: management of access policy, where to place firewalls, ...
- SCADA (Supervision Control And Data Acquisition)
  In automotive domain:
- EVITA: E-safety vehicle intrusion protected applications (uses attack trees)
- SAEJ3061: The new automotive cybersecurity standard (2016)

Problems:

- Exhaustivity of the attack tree.
- Complex to draw when the system to analyze is large.
- Error prone: Security experts have to imagine all possible ways

Automatic generation of attack trees

Solution: automatic generation of attack trees

Given a system model and an adversary model we can see if the adversary can put the system in a vulnerable state and trace its actions.

Advantages:

- Addresses all attack vectors in a structured way.
- Automatically analyze the vehicle architecture (however large it may be).
- Lists all possible attack paths w.r.t the system and adversary model: exhaustivity.

Graph transformation

Graph transformation

- The modeling is based on the graph transformation, which allows us to express a relational model in the form of a graph (vertex and arcs) and to explore the different possible configurations of this graph by applying transformations rules:
  - Start Graph: System model
  - Transformation rules: inference rules to automate a reasoning method (deduction/derivation)
- tool: Groove $\rightarrow$ is a tool for Graph transformation.

Model:

Modeled components: graph

- Communication nodes
- Hardware nodes: communication controllers and data storage.
- Service nodes and access rights to hardware components.
- Data nodes.

Behavioral model: transformation rules

- Behavior of hardware nodes.
- Behavior of service nodes.
- Behavior of the attacker.

Model:

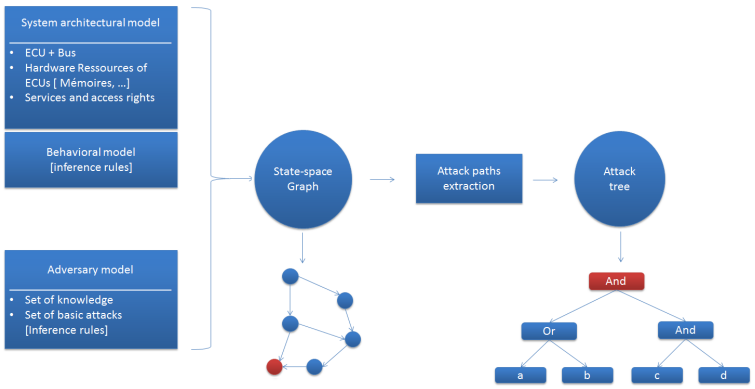


Figure : Automatic generation of attack trees

Illustration: model

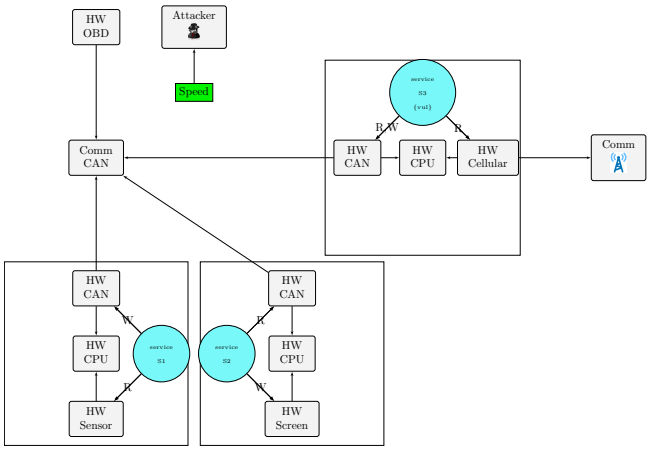


Figure : Speed acquisition and display

Transformation rules:

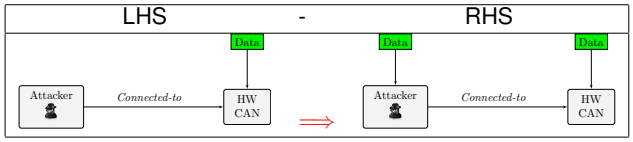


Figure : Transformation rule: attacker eavesdrop on CAN

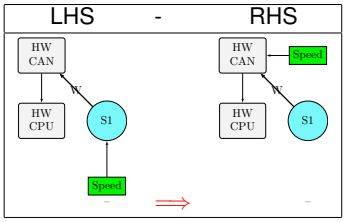


Figure : Transformation rule: Speed send to CAN

Illustration: model

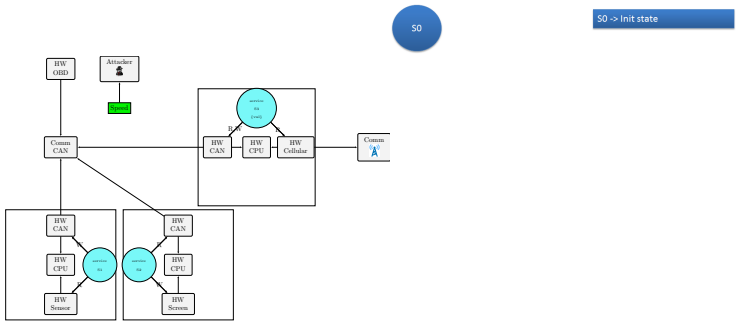


Illustration: model

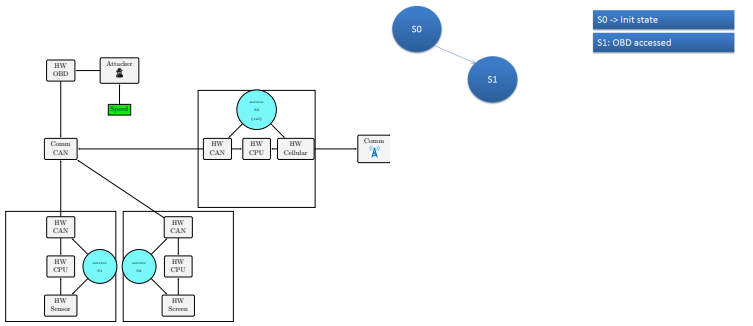


Illustration: model

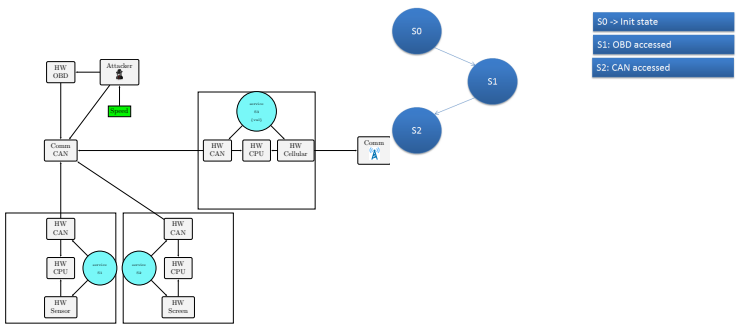
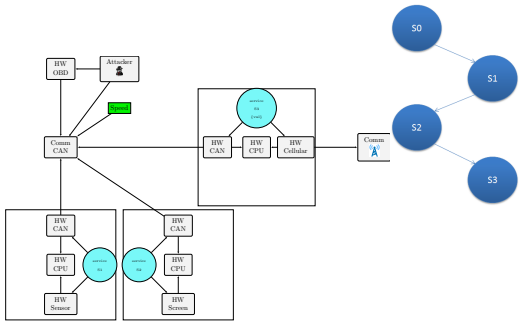
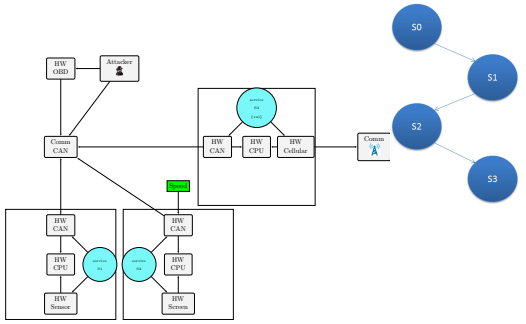


Illustration: model



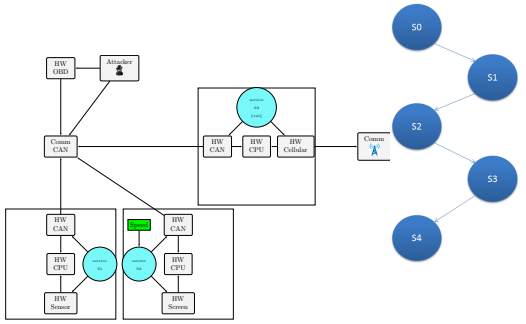
- S0 -> Init state
- S1: OBD accessed
- S2: CAN accessed
- S3: Speed injected on CAN

Illustration: model



- S0 -> Init state
- S1: OBD accessed
- S2: CAN accessed
- S3: Speed injected on CAN

Illustration: model



- S0 -> Init state
- S1: OBD accessed
- S2: CAN accessed
- S3: Speed injected on CAN
- S4: Speed read by S2

Illustration: model

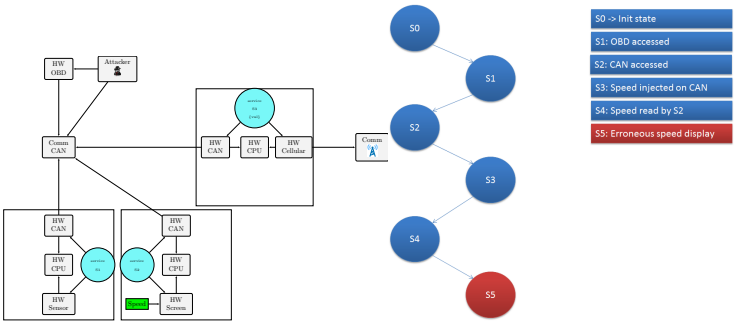


Illustration: Attack tree

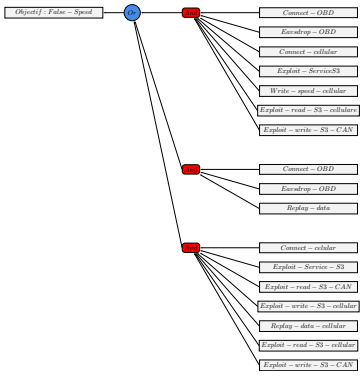


Figure : Attack tree: Speed acquisition and display

Illustration: model

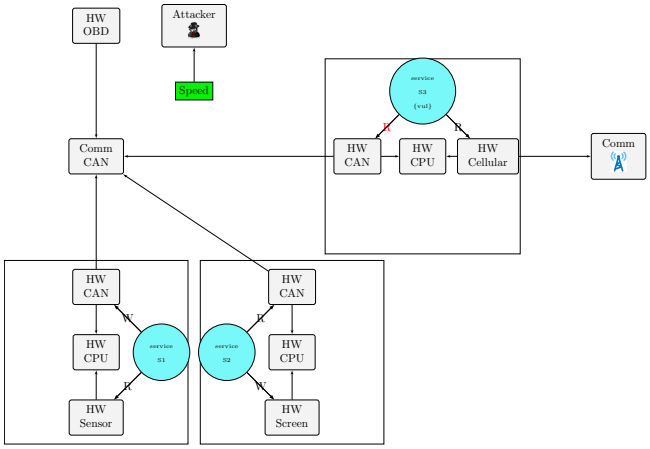


Figure : Speed acquisition and display

Illustration: Attack tree - countermeasure

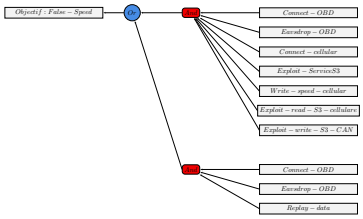


Figure : Attack tree: Speed acquisition and display

Plan

Conclusion:

- We need to consider security from design phase in order to build secure cars.
- We need to properly estimate the risk associated with threats.
- Attack trees are good tool for security risk assessment.
- Automatic generation of attack trees to overcome possible human errors and for exhaustivity.

Thank you for your attention!

Demonstration of the Acoustic Cryptanalysis

Tomas Fabsic, Ondrej Gallo, Viliam Hromada

Slovak University of Technology in Bratislava

CryptArchi 2017

Contents

Contents

About the attack

- We repeated the attack described in:
  Genkin, D., Shamir, A., Tromer, E. (2014). *RSA key extraction via low-bandwidth acoustic cryptanalysis.*
- Source of the exploited acoustic signal: coils and capacitors within the laptop's voltage regulator.

Contents

Attacked laptop

- The type of laptop matters, not every laptop can be attacked.
- We used Lenovo ThinkPad T61 laptop.
- The laptop ran the RSA implementation from GnuPG 1.4.14..

Microphone

- Needs to be sensitive above acoustic spectrum. (40kHz in case of our laptop)
- We used equipment from Bruel&Kjaer:
  - microphone 4190
  - preamplifier 2669
  - amplifier and power supply 5935

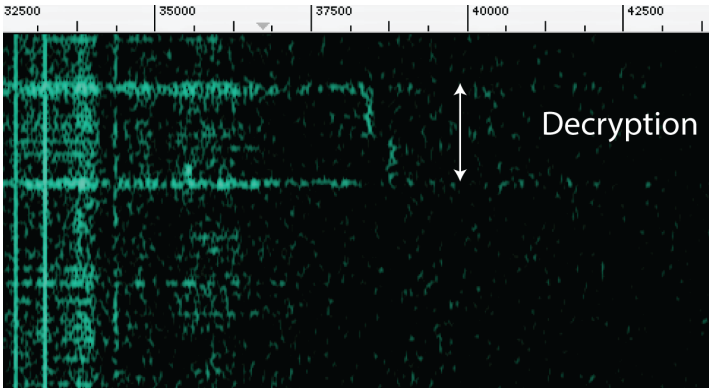
Computer for signal analysis

- A personal computer for analysis of the recorded acoustic signal.
- Contained the baudline signal analysis tool (freeware).

Contents

RSA decryption in GnuPG 1.4.14.

- Uses the Chinese remainder theorem:
  - Firstly $y^d \mod p$ is computed.
  - Afterwards $y^d \mod q$ is computed.
- How does the frequency spectrum of the acoustic signal during decryption look like?



Contents

- There is a relationship between position of lines in the spectrogram and the value of the secret primes.
- This can be exploited to reveal the value of the secret primes.

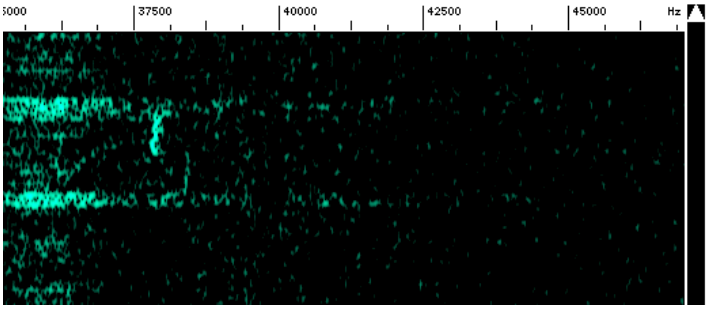
- We can do an adaptive chosen ciphertext attack.
- We reveal the value of the secret prime q bit by bit.
- We know the length of q.
- We know q starts with 1.

To reveal the second bit of q we:

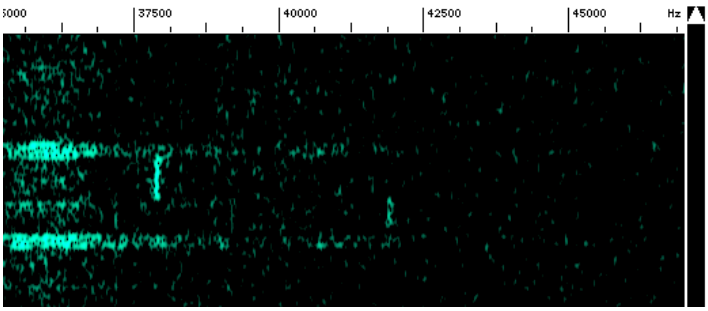
- Make the laptop decrypt the ciphertext $y = 1011\,1111\,1111\,1111\,1111\ldots$, where y has the same length as q.
- The decryption algorithm firstly computes $c = y \mod q$ and exponentiates c instead of y.
- In case the second bit of q is 0, c has a "random looking" structure $\Rightarrow$ ordinary pattern in the spectrogram.
- In case the second bit of q is 1, c has a "very special" structure $\Rightarrow$ strange pattern in the spectrogram.
- Observing the spectrogram, we can detect scenario occurred and determine the value of the second bit.

After learning the second bit, we construct a new ciphertext to reveal the third bit in the same fashion...

Spectrogram when the second bit is 0:



Spectrogram when the second bit is 1:



The End

Thank you for your attention!

Lessons Learned from High-Speed Implementation and Benchmarking of Two Post-Quantum Public-Key Cryptosystems

Malik Umar Sharif,
Ahmed Ferozpuri, and
Kris Gaj
George Mason University
USA

1

Post-Quantum Cryptography

- Quantum Computers could potentially break all current American federal standards in the area of public-key cryptography (RSA, ECC, Diffie-Hellman)
- Increasing key sizes would be futile
- Search for new public key cryptographic families resistant against quantum computing cryptanalysis, collectively referred to as Post-Quantum Cryptography (PQC)
- PQC algorithms capable of
  - being implemented using any traditional methods, including software and hardware
  - running efficiently on any modern computing platforms: PCs, tablets, smartphones, servers with hardware accelerators, etc.

2

Post-Quantum Cryptography Efforts

- New public-key cryptographic families: mid-1990s-present
- Series of PQCrypto Conferences: 2006-present
- NIST Workshop on Cybersecurity in a Post-Quantum World 2015
- NIST announcement of standardization plans at PQCrypto 2016: Feb. 2016
- NIST Call for Proposals and Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms: Dec. 2016

  Deadline for submitting candidates: November 30, 2017
- Time of Standard Development + Time of Standard Deployment + Max. Protection Time
  must be smaller than
  Time to Develop Sufficiently Large Quantum Computer

3

Promising PQC Cryptoalgorithm Families

Family	Encryption	Signature	Key Agreement
Hash-based		X	
Code-based	X		
Lattice-based	X	X	
Multivariate	X	X	
Supersingular Elliptic Curve Isogeny			X

4

Our Objectives

Paving the way for the future comprehensive, fair, and efficient hardware benchmarking of PQC candidates through

1. Uniform Hardware API
2. Uniform & Efficient Development Process based on
   a. detailed flow diagrams
   b. choice of supported parameter sets
   c. top-level & lower-level block diagrams
   d. cycle-based timing analysis
   e. Algorithmic State Machine (ASM) charts
   f. Register-Transfer Level (RTL) code
   g. software-generated test vectors,
   h. comprehensive testbenches,
   i. results of synthesis and implementation
   j. analysis of results & lessons learned

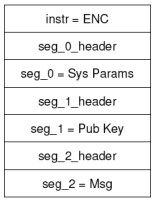
5

Proposed Uniform Hardware API

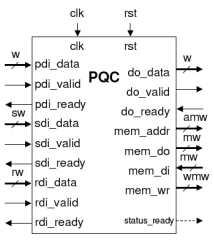
Specifies:

- Minimum Compliance Criteria
- Interface
- Communication Protocol
- Timing Characteristics

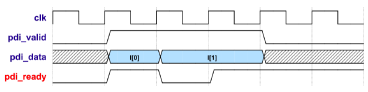
Communication Protocol: Format of Input Data



Interface



Timing Characteristics



6

Algorithms Selected for a Pilot Study

1. NTRUEncrypt Short Vector Encryption Scheme (SVES)
   fully compliant with
   IEEE 1363.1 Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices: 2008
   & Efficient Embedded Security (EES) Standard: 2015

   Parameter sets:
   - ees1087ep1 and ees1499ep1
   - 192 and 256-bit security level
   - Optimized for speed

2. Multivariate Rainbow Signature Scheme

   Parameter set:
   - (17,12)(1,12)
   - 80-bit security level

7

NTRUEncrypt – Core Functionality (1)

Parameters:
N – prime
p – small prime, typically 3
q – power of 2, typically 2048

Basic Operations:
Polynomial Multiplication, Addition, Subtraction in the ring $Z/qZ[X]/X^N-1$

Private Key:
f = 1+pF, where F – random polynomial with small coefficients {-1, 0, 1}

Public Key:
$h = f^{-1} * g \bullet p$, and
g – random polynomial with small coefficients {-1, 0, 1}

8

NTRUEncrypt – Core Functionality (2)

Encryption:

$$e = r * h + m \pmod q$$

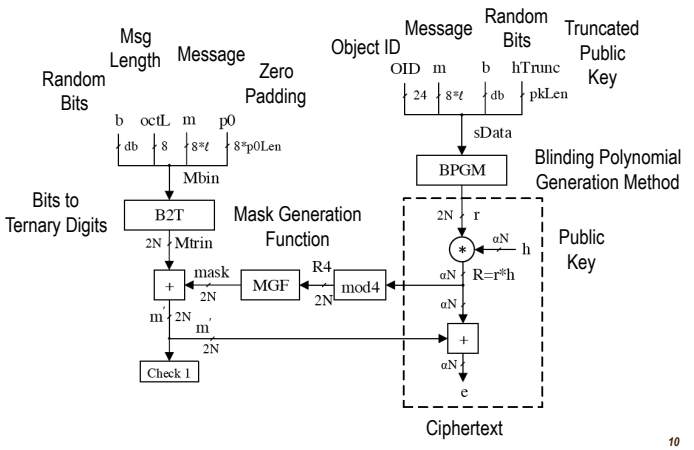
where r is a random polynomial with small coefficients

Decryption:

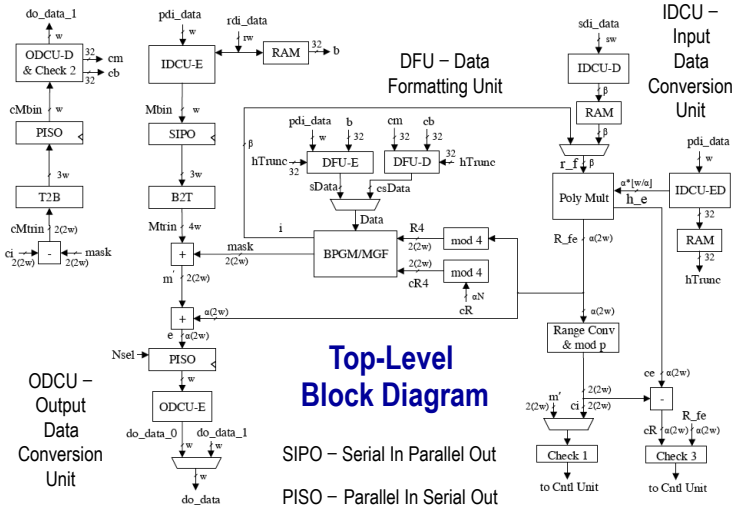
1) calculate $f * e \pmod q$
2) shift coefficients of the obtained polynomial to the range $[-q/2, q/2)$,
3) reduce the obtained coefficients mod p

9

NTRUEncrypt – Flow Diagram for Encryption



10

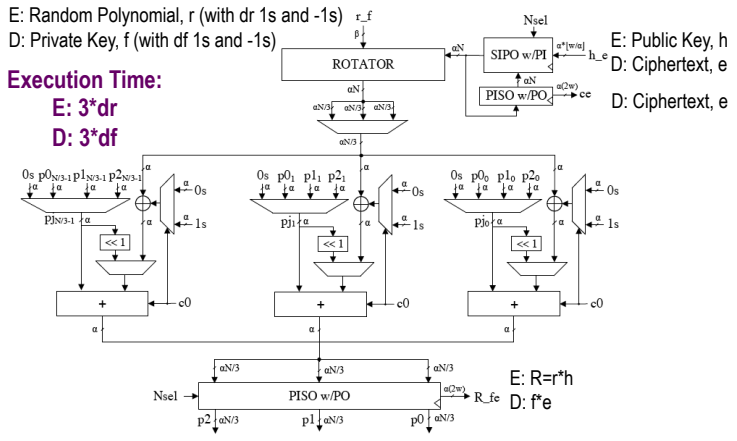


11

Block Diagram of Polynomial Multiplier

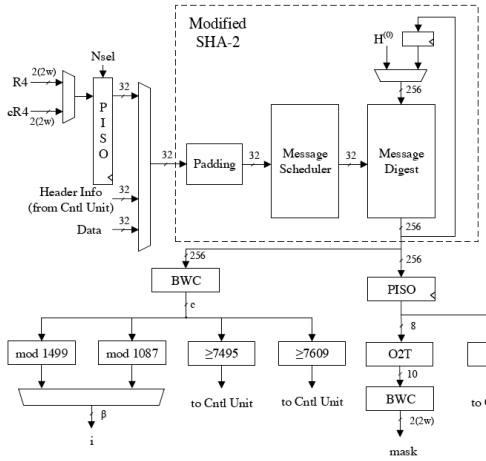
E: Random Polynomial, r (with dr 1s and -1s)
D: Private Key, f (with df 1s and -1s)

Execution Time:
E: 3*dr
D: 3*df



12

Block Diagram of Blinding Polynomial Generation Method / Mask Generation Function



BWC – Bit Width Conversion

O2T – Octets to Ternary Digits

PISO – Parallel In Serial Out

Implementation of SHA-2 modified to share computations for hashing of strings starting from the same common substring

13

Major Component Operations Resource Utilization & Performance

Operation	LUTs : Slices	Clk Freq [MHz]	Latency [cycles]	Latency·LUTs
Poly Mult	138,475 : 55,535	89.51	474	6,387,150
BPGM	2519 : 766	148.28	845	2,128,555
MGF			1004	2,529,076
B2T	104 : 55	734.75	1	104
T2B	198: 103	495.04	1	198
Poly Adds e=(Mtrin+mask)+R_fe	1868 : 718	264.55	1	1868
Poly Sub cMtrin=ci-mask	342 : 127	572.41	1	342

14

Comparison with Previous Work on Implementing Polynomial Multiplication

Source	Resources	Clk Freq [MHz]	Latency [cycles]	Latency [µs]
Parameter set: ees1499ep1				
Liu et al. [22]	83,949 LEs	63.64	867	13.62
This Work	138,475 LUTs	89.51	474	5.30
	Speed-up	x1.41	x1.83	x2.57
Parameter set: ees1087ep1				
Liu et al. [22]	60,876 LEs	73.71	638	8.65
This Work	138,475 LUTs	89.51	378	4.22
	Speed-up	x1.21	x1.69	x2.05

B. Liu and H. Wu, "Efficient Multiplication Architecture over Truncated Polynomial Ring for NTRUEncrypt System," IEEE International Symposium on Circuits and Systems, ISCAS 2016

15

Profiling Software Implementation on ARM Cortex A9

Software Function	Hardware Equivalent	Clock cycles	% of Total Time
ntru_gen_poly	Performing BPGM on sData & calculating R using Poly Mult (in a pipelined fashion)	24,779	2.3%
ntru_octets_2_elements		12,728	1.2%
ntru_ring_mult_product_indices		950,892	89.4%
ntru_coeffs_mod4_2_octets	Calculating cR4 using mod 4 & mask using MGF	9,427	0.9%
ntru_mgftp1		30,703	2.9%
ntru_bits_2_trits		3,020	0.3%
adding Mtrin to mask	Calculating m' using Poly Add & performing Check 1	8,108	0.8%
ntru_poly_check_min_weight		6,910	0.6%
add_m'		8,672	0.8%
elements_2_octets	Unloading ciphertext e	13,549	1.3%
Total		**1,068,788**	**100.0%**

16

Profiling Hardware Implementation on Xilinx Virtex-7

Operation	Latency (clock cycles)	% of Total Time	Latency (clock cycles)	% of Total Time
	ees1499ep1		ees1087ep1	
ENCRYPTION				
Performing BPGM on sData & calculating R using Poly Mult (in a pipelined fashion)	890	38.8%	701	39.5%
Calculating cR4 using mod 4 & mask using MGF	1005	43.8%	787	44.3%
Calculating m` using Poly Add & performing Check 1	97	4.2%	70	3.9%
Unloading ciphertext e	300	13.1%	218	12.3%
Total	**2292**	100%	**1776**	100%

17

Hash Function Bottleneck

Software

- **Poly Mult amounts to about 90% of the total execution time**

Hardware

- **Execution time dominated by hash-based**
  - **MGF: Mask Generation Function: 44%**
  - **BPGM: Blinding Polynomial Generation Method: 39.5%**
- **Poly Mult almost completely overlapped with the computations of BPGM through the use of pipelining**
- **Poly Mult naturally parallelizable**
- **Hash function naturally sequential**

18

Possible Improvements

To Address the Hash Function Bottleneck:

Architecture-Level:

- Unrolled Implementation of SHA-2

Algorithmic-Level (changes in the IEEE and EES standards required):

- SHA-3 instead of SHA-2
- Pseudorandom function based on the pipelined AES

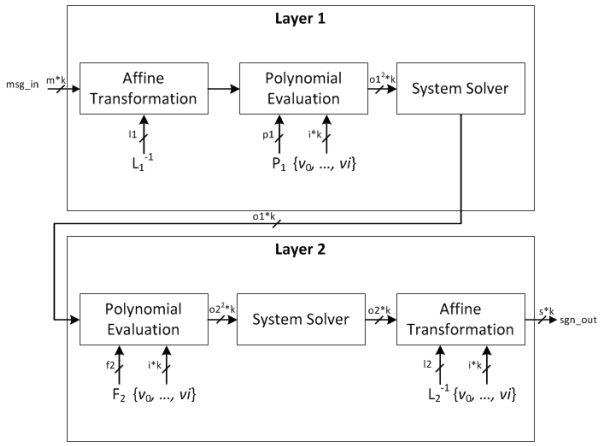
To Address Other Encountered Problems:

Algorithmic-Level (changes in the IEEE and EES standards required):

- Using SHA-2 instead of SHA-1 in lower-security parameter sets
- Eliminating the dependence of the execution time on message size

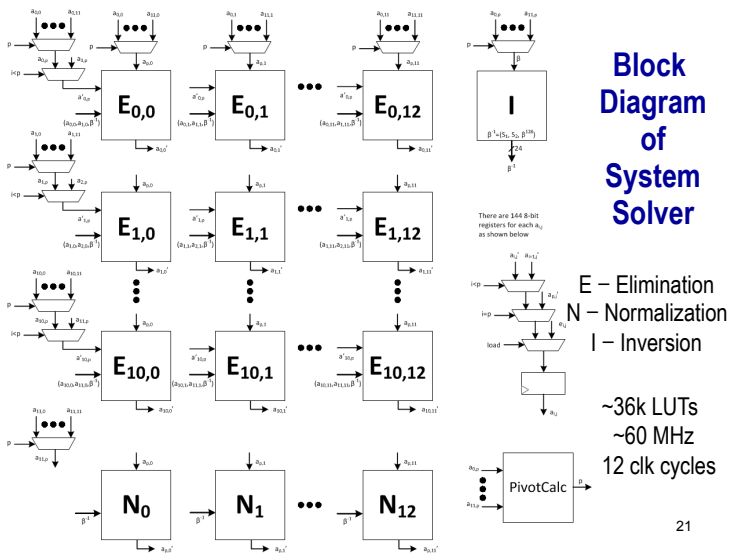
19

Rainbow: Flow Diagram for Signature Generation



20

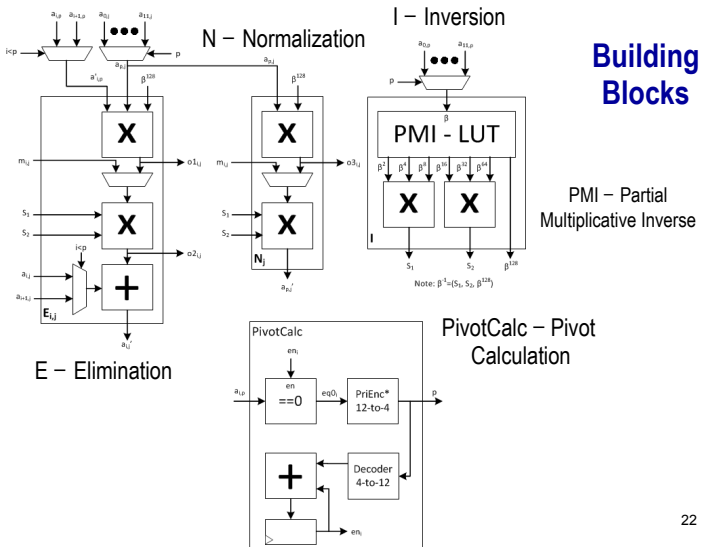
Block Diagram of System Solver



E – Elimination
N – Normalization
I – Inversion

~36k LUTs
~60 MHz
12 clk cycles

21



22

NTRUEncrypt vs. Rainbow Signature Scheme

Security Levels:

NTRU: **Parameter sets supporting 112, 128, 192, & 256 bit security levels**
Rainbow: **Published parameter sets at 80-90 bit security levels**

Key Sizes:

	Security Level	Public Key Size	Private Key Size
NTRU	192	1495 B	87 B
	256	2062 B	109 B
Rainbow	80	19.64 kB	30.94 kB

23

Comparative Analysis of Implementation Difficulties

Feature	NTRUEncrypt	Rainbow SS
High-security levels	Easy to implement	Challenging to implement
Key sizes	Small	Very Large
Support for multiple parameter sets swapped at run time	Relatively easy to implement	Challenging to implement
Component operations	Standard: variable rotator, hash function	Complex: System of Linear Equation Solver
Dependence of the execution time on message size	Strong	Weak

24

Hardware Architectures for HECC

Gabriel GALLIN and **Arnaud TISSERAND**

CNRS – Lab-STICC – IRISA
HAH Project

CryptArchi June, 2017

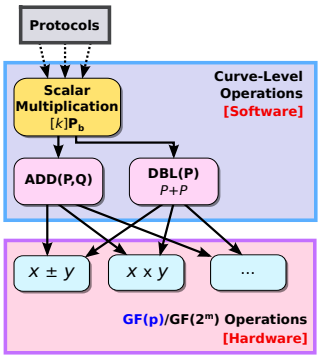
Summary

Summary

Public-Key Cryptography (PKC)

- Provides cryptographic primitives such as digital signature, key exchange and specific encryption schemes
- First PKC standard: RSA
  - Large keys ($\geq$ 2000 bits recommended today)
  - Too costly for embedded applications
- Elliptic Curve Cryptography (ECC):
  - Better performances and lower cost than RSA
  - Allows more advanced schemes
- Hyper-Elliptic Curve Cryptography (HECC):
  - Evolution of ECC focusing on larger sets of curves
  - Studied for future generations of asymmetric crypto-systems

Operations Hierarchy in (H)ECC



- ADD and DBL built using $\mathbb{F}_\mathcal{P}$ operations
- Modular arithmetic in $\mathbb{F}_\mathcal{P}$:
  - Size of elements: 100 $\sim$ 200 bits for HECC
  - Operations involve modular reduction
  - Choice of $\mathcal{P}$:
    - Generic $\mathcal{P}$: more flexible but slower
    - Specific $\mathcal{P}$ (*e.g.* pseudo-Mersenne): faster but more specific
- Modular multiplication (M) and square (S):
  - Most common and costly operations
  - Dedicated arithmetic unit(s) must be efficient

Metric: mainly numbers of multiplications (M) and squares (S) in $\mathbb{F}_\mathcal{P}$

ECC, HECC, Kummer-HECC

	size of $\mathbb{F}_\mathcal{P}$	ADD	DBL	source
ECC	ℓ_{ECC}	12M + 2S	7M + 3S	[Bernstein and Lange]
HECC	$\ell_{\text{HECC}} \approx \frac{1}{2}\ell_{\text{ECC}}$	40M + 4S	38M + 6S	[Lange, 2005]
Kummer	ℓ_{HECC}	19M + 12S		[Renes et al., 2016]

- ECC:
  - Size of $\mathbb{F}_\mathcal{P}$ elements 2$\times$ larger
  - Simpler ADD and DBL operations
- HECC:
  - Smaller $\mathbb{F}_\mathcal{P}$
  - More operations in $\mathbb{F}_\mathcal{P}$ for ADD and DBL
- Kummer-HECC is more efficient than ECC, see [Renes et al., 2016]:
  - ARM Cortex M0: up to 75% clock cycles reduction for signatures
  - AVR AT-mega: up to 32% cycles reduction for Diffie-Hellman

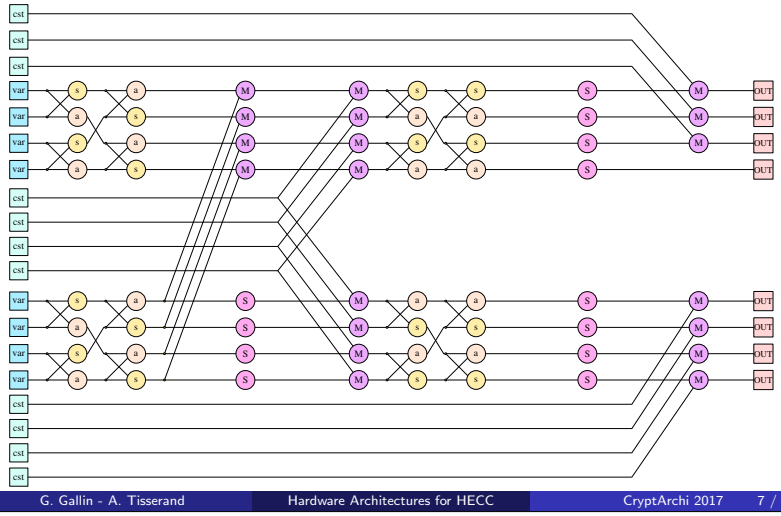
Summary

Curve-Level Operations in Kummer

- No ADD operation but still DBL
- *Differential addition* operation:
  $\texttt{xADD}(\pm P, \pm Q, \pm(P-Q)) \rightarrow \pm(P+Q)$
- xADD and DBL can be combined:
  $\texttt{xDBLADD}(\pm P, \pm Q, \pm(P-Q)) \rightarrow (\pm[2]P, \pm(P+Q))$

For details see [Renes et al., 2016], [Gaudry, 2007] and [Bos et al., 2016]

xDBLADD $\mathbb{F}_{\mathcal{P}}$ Operations



Scalar Multiplication

Montgomery ladder based *crypto_scalarmult* [Renes et al., 2016]:

Require: m-bit scalar $k = \sum_{i=0}^{m-1} 2^i k_i$, point P_b, $cst \in \mathbb{F}_{\mathcal{P}}^4$
Ensure: $V_1 = [k]P_b$, $V_2 = [k+1]P_b$
  $V_1 \leftarrow cst$
  $V_2 \leftarrow P_b$
  for $i = m-1$ **downto** 0 **do**
    $(V_1, V_2) \leftarrow \texttt{CSWAP}(k_i, (V_1, V_2))$
    $(V_1, V_2) \leftarrow \texttt{xDBLADD}(V_1, V_2, P_b)$
    $(V_1, V_2) \leftarrow \texttt{CSWAP}(k_i, (V_1, V_2))$
  end for
  return (V_1, V_2)

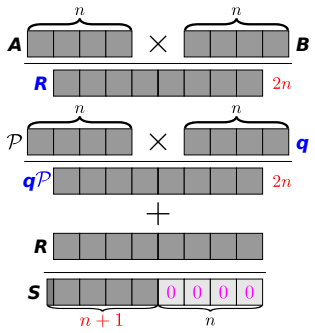
$\texttt{CSWAP}(k_i, (X, Y))$ returns (X, Y) if $k_i = 0$, else (Y, X)

- Constant time, uniform operations (independent from key bits)
- Some parallelism between xDBLADD internal $\mathbb{F}_{\mathcal{P}}$ operations
- CSWAP: very simple but involves secret bits (to be protected)

Summary

Montgomery Modular Multiplication (MMM) Algorithm

$$R = A \times B \qquad n \times n \rightarrow 2n \text{ bits}$$
$$q = (R \times (-\mathcal{P}^{-1})) \mod (2^n) \qquad n \times n \rightarrow n \text{ bits}$$
$$q\mathcal{P} = q \times \mathcal{P} \qquad n \times n \rightarrow 2n \text{ bits}$$



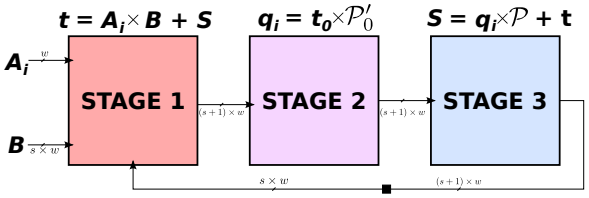
- Proposed in [Montgomery, 1985]
- Variants are actual state-of-the-art for $\mathbb{F}_{\mathcal{P}}$ multiplication (with generic $\mathcal{P}$)
- Final reduction step by discarding n LSBs
- Multiplication and reduction steps can be interleaved

Modular Multiplication: Dependencies Problem

- In practice, MMM is interleaved
  - Operands are split into s words of w bits such that $n = s \times w$
  - MMM consists in iterations over partial products and reductions
  - *Coarsely Integrated Operand Scanning* (CIOS) from [Koç et al., 1996]
- Impact on hardware implementation
  - Dependencies → latencies between internal iterations
  - Hardware pipeline in DSP slices cannot be filled efficiently
- Proposed solution: Hyper-Threaded Modular Multiplier (HTMM)
  - Based on CIOS algorithm
  - Use idle hardware to compute other independent MMMs in parallel

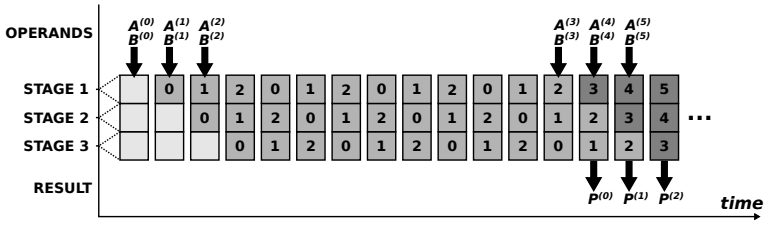
HTMM Internal Architecture

- HTMM architecture: 3 hardware stages
  - Stages are fully pipelined (several clock cycles per stage)
  - 3 to 4 DSP slices in each stage



HTMM Internal Architecture

- HTMM architecture: 3 hardware stages
  - Stages are fully pipelined (several clock cycles per stage)
  - 3 to 4 DSP slices in each stage



HTMM Implementations

- Implementations on 3 Xilinx FPGAs
  - Virtex 4 XC4VLX100 (V4)
  - Virtex 5 XC5VLX110T (V5)
  - Spartan 6 XC6SLX75 (S6)
- Comparison with fastest MMM implementation in literature
  - Design presented in [Ma et al., 2013]
  - Implemented on the same FPGAs for fair comparison
- 2 versions of HTMM:
  - HTMM_DRAM : operands stored in FPGA slices (LUTs)
  - HTMM_BRAM : operands stored in FPGA BRAMs
- Parameters for HTMM:
  - $\mathcal{P} \rightarrow$ 128 bits
  - $w = 34$ bits, $s = 4$
  - Operands size $n = s \times w = 134$ bits

HTMM Implementations Results

Results for 3 independent multiplications:

Version	FPGA	DSP	BRAM 18K/9K	FF	LUT	Slices	Freq. (MHz)	nb. cycles	Time (ns)
[Ma et al., 2013]	V4	21	6*/0	1311	1201	879	252	65	258
	V5	21	6*/0	1310	1027	406	296		220
	S6	21	0/6*	1280	1600	540	210		309
HTMM_DRAM	V4	11	0/0	1638	1128	1346	330	79	239
	V5	11	0/0	1616	652	517	400		198
	S6	11	0/0	1631	1344	483	302		261
HTMM_BRAM	V4	11	2/0	615	364	449	328	79	241
	V5	11	2/0	593	371	249	357		221
	S6	11	0/2	587	359	180	304		260

For only 1 single `M`, HTMM is less efficient (69 cycles against 25)

Summary

Architectures Exploration for (H)ECC

- Constraints:
  - Coding a complete accelerator and its units in HDL is costly
  - Large solution space for various architectures types and parameters (nb. units, algorithms, internal communications and control)
  - Need for evaluation of various architectures and parameters then select interesting solutions
  - Need for numerical validation and debug
- Proposed solution:
  - Hierarchical description and simulation at CCABA level (Critical-Cycle Accurate, Bit Accurate)
  - Units described in HDL with perfectly known behavior
  - High-level architecture description supporting many models and parameters

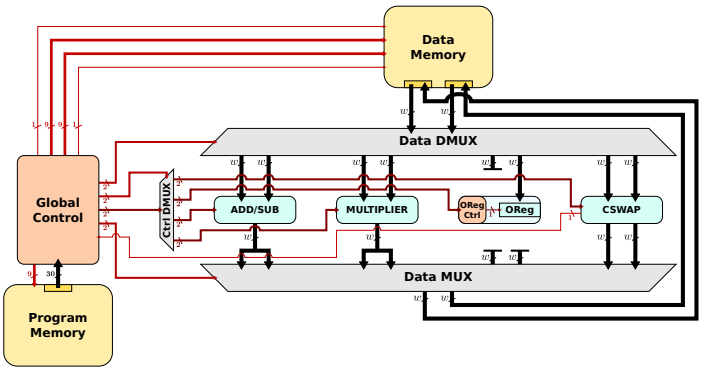
A Simulator to Explore Architectures

- We developed a simulator dedicated to HECC architectures
- Description and simulation at CCABA level
  - Control signals and communications between units are fully accurate
  - A scalar multiplication takes several thousand cycles
  - Only critical cycles in units (inputs, outputs and control) are used in the high-level architecture model
- Simulator coded in Python
  - Fast development
  - Provides interface with Sage (`http://www.sagemath.org/`) tool for numerical validation and HECC support

Modeling of Architectures

- Architecture models built over a set of units
  - Arithmetic units: adders, subtractors, multipliers, ...
  - Memories, registers, ...
  - Interconnect: buses, multiplexers, ...
- Units models:
  - All inputs/outputs are bit accurate
  - All inputs/outputs and control signals are critical cycle accurate
  - Units characteristics come from FPGA implementation (latency, area cost, ...)
- Control from curve operations formulas and units configurations
  - Manage activity of units in the architecture
  - Manage all internal communications
  - Developed tool for scheduling $\mathbb{F}_p$ operations (*work in progress*)

Architecture Model



Parameters specified at design time:

- Width w and nb. words s for internal communications ($s \times w = n$)
- Types and number of units

Functional Units Configuration

- $\mathbb{F}_\mathcal{P}$ adder-subtractor unit (ADD/SUB):
  - One unit for two types of operation
  - Pipelined operator (4 cycles latency)
  - Delay: 8 $\sim$ 11 cycles depending on external datapath width
- $\mathbb{F}_\mathcal{P}$ multiplier unit (HTMM):
  - Hyper-threaded multiplier: 3 sets of operands computed in parallel
  - Delay: 68 $\sim$ 71 cycles depending on external datapath width
  - Latency: 5 cycles (for loading and reading)
- CSWAP unit:
  - Secure management of key bits
  - Delay: 2 $\sim$ 4 cycles depending on w

Results for the Basic Architecture (1 Add/Sub, 1 HTMM)

Version ($s \times w$)*	Number of cycles	Functional Unit	DSP	BRAM	FF	LUT	Slices	RAM size (nb. lines)
4x34	207383	HTMM	11	2	587	359	180	12
		ADDSUB	0	0	366	226	80	-
		DATA_MEM	0	1	0	0	0	112
		PRGM_MEM	0	1	0	0	0	208
		CSWAP	0	0	536	290	103	-
2x68	185615	HTMM	11	2	970	633	315	12
		ADDSUB	0	0	713	382	148	-
		DATA_MEM	0	2	0	0	0	56
		PRGM_MEM	0	1	0	0	0	234
		CSWAP	0	0	553	297	122	-
1x136	183051	HTMM	11	2	1066	623	309	12
		ADDSUB	0	0	784	464	212	-
		DATA_MEM	0	4	0	0	0	26
		PRGM_MEM	0	1	0	0	0	250
		CSWAP	0	0	685	431	155	-

* s: number of words, w: size of words

Increasing the Number of Arithmetic Units

Version ($s \times w$)*	Number of cycles	Functional Unit	DSP	BRAM	FF	LUT	Slices	RAM size (nb. lines)
4x34	**203543**	HTMM **x 2**	**22**	**4**	**1174**	**718**	**360**	12
		ADDSUB **x 2**	0	0	**732**	**452**	**160**	-
		DATA_MEM	0	1	0	0	0	**108**
		PRGM_MEM	0	1	0	0	0	**213**
		CSWAP	0	0	536	290	103	-
2x68	**125455**	HTMM **x 2**	**22**	**4**	**1940**	**1266**	**630**	12
		ADDSUB **x 2**	0	0	**1426**	**764**	**296**	-
		DATA_MEM	0	4	0	0	0	**50**
		PRGM_MEM	0	1	0	0	0	**211**
		CSWAP	0	0	553	297	122	-
1x136	**115211**	HTMM **x 2**	**22**	**4**	**2132**	**1246**	**618**	12
		ADDSUB **x 2**	0	0	**1568**	**928**	**424**	-
		DATA_MEM	0	4	0	0	0	**25**
		PRGM_MEM	0	1	0	0	0	**235**
		CSWAP	0	0	685	431	155	-

* s: number of words, w: size of words

256b ECC vs 128b HECC (similar theoretical security)

FPGA	Version	DSP	BRAM 36K/18K	Slices	Freq. (MHz)	nb. cycles	Time (ms)
V4	ECC	37	0/11	4655	250	109297	0.44
	HECC_1u	11	0/7	1413	330	183051	0.55
	HECC_2u	22	0/9	2356	330	115211	0.35
V5	ECC	37	10/0	1725	291	109297	0.38
	HECC_1u	11	0/7	873	360	183051	0.51
	HECC_2u	22	0/9	1542	360	115211	0.32

ECC results from [Ma et al., 2013]

Conclusions and Perspectives

- Kummer based HECC is an efficient alternative to ECC
  - More complex formulas but larger internal parallelism
  - Large exploration space for architectures and arithmetic
- We designed a CCABA simulator and modeling
  - High-level hierarchical description of architectures
  - Units described in HDL, only critical cycles are used
  - Fast validation/debug and evaluation of solutions in exploration space
- Perspectives and future work
  - Study advanced scheduling algorithms
  - Automating generation of HDL code from high-level description
  - Explore new architectural solutions

This work is funded by **HAH** project http://h-a-h.inria.fr/

Thank you for your attention

References I

[Bernstein and Lange] Bernstein, D. J. and Lange, T.
Explicit-formulas database.
http://hyperelliptic.org/EFD/.

[Bos et al., 2016] Bos, J. W., Costello, C., Hisil, H., and Lauter, K. (2016).
Fast cryptography in genus 2.
Journal of Cryptology, 29(1):28–60.

[Cohen et al., 2005] Cohen, H., Frey, G., Avanzi, R., Doche, C., Lange, T., Nguyen, K., and Vercauteren, F. (2005).
Handbook of Elliptic and Hyperelliptic Curve Cryptography.
Discrete Mathematics and Its Applications. Chapman & Hall/CRC.

[Gaudry, 2007] Gaudry, P. (2007).
Fast genus 2 arithmetic based on theta functions.
Journal of Mathematical Cryptology, 1(3):243–265.

[Hankerson et al., 2004] Hankerson, D., Menezes, A., and Vanstone, S. (2004).
Guide to Elliptic Curve Cryptography.
Springer.

[Koç et al., 1996] Koç, Ç. K., Acar, T., and Kaliski, Jr., B. S. (1996).
Analyzing and comparing Montgomery multiplication algorithms.
Micro, IEEE, 16(3):26–33.

[Lange, 2005] Lange, T. (2005).
Formulae for Arithmetic on Genus 2 Hyperelliptic Curves.
Applicable Algebra in Engineering, Communication and Computing, 15(5):295–328.

References II

[Ma et al., 2013] Ma, Y., Liu, Z., Pan, W., and Jing, J. (2013).
A high-speed elliptic curve cryptographic processor for generic curves over GF(p).
In *Proc. 20th International Workshop on Selected Areas in Cryptography (SAC)*, volume 8282 of *LNCS*, pages 421–437.
Springer.

[Montgomery, 1985] Montgomery, P. L. (1985).
Modular multiplication without trial division.
Mathematics of Computation, 44(170):519–521.

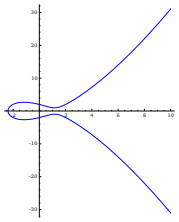
[Montgomery, 1987] Montgomery, P. L. (1987).
Speeding the Pollard and elliptic curve methods of factorization.
Mathematics of Computation, 48(177):243–264.

[Orup, 1995] Orup, H. (1995).
Simplifying quotient determination in high-radix modular multiplication.
In *Proc. 12th Symposium on Computer Arithmetic (ARITH)*, pages 193–199. IEEE Computer Society.

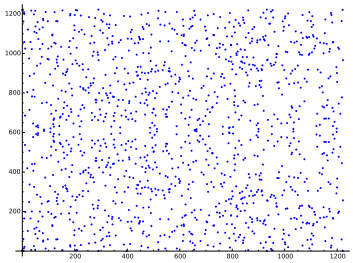
[Renes et al., 2016] Renes, J., Schwabe, P., Smith, B., and Batina, L. (2016).
μKummer: Efficient hyperelliptic signatures and key exchange on microcontrollers.
In *Proc. Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, volume 9813 of *LNCS*, pages 301–320.
Springer.

Elliptic and Hyper-Elliptic Curves for Crypto

- Elliptic Curves
  - Equation (Weierstrass) $E/\mathbb{K} : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$
  - Defined over finite fields $\mathbb{K}$: $\mathbb{F}_{2^m}$, prime finite field $\mathbb{F}_\mathcal{P}$ or $GF(p)$
  - $\mathbb{F}_\mathcal{P}$ elements for coefficients and coordinates: 200 $\sim$ 400 bits



Curve over $\mathbb{R}$ (not for crypto)



Curve over $\mathbb{F}_{1223}$

Summary Context & Motivations HECC Operations Arithmetic Units Architectures and Tools Conclusion

Elliptic and Hyper-Elliptic Curves for Crypto

- Elliptic Curves
  - Equation (Weierstrass) $E/\mathbb{K} : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$
  - Defined over finite fields $\mathbb{K}$: $\mathbb{F}_{2^m}$, prime finite field $\mathbb{F}_\mathcal{P}$ or $GF(p)$
  - $\mathbb{F}_\mathcal{P}$ elements for coefficients and coordinates: 200 $\sim$ 400 bits

- Hyper-Elliptic Curves
  - **More complex!**
  - Equation $H/\mathbb{K} : y^2 + h(x)y = f(x)$, $deg(h) < g$ and $deg(f) = 2g + 1$
  - g: genus of the curve, $g \leq 2$ in practice for reliable HECC
  - $\mathbb{F}_\mathcal{P}$ elements for coefficients and coordinates: 100 $\sim$ 200 bits

Summary Context & Motivations HECC Operations Arithmetic Units Architectures and Tools Conclusion

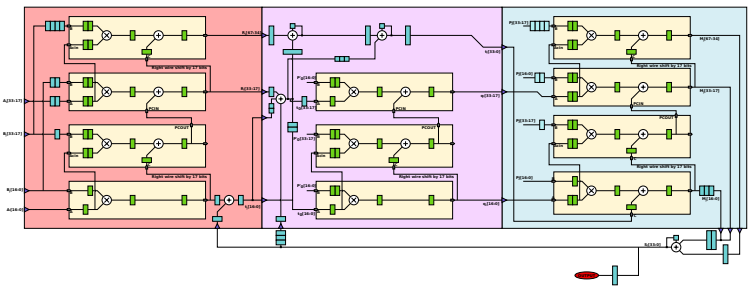
Elliptic and Hyper-Elliptic Curves for Crypto

- Elliptic Curves
  - Equation (Weierstrass) $E/\mathbb{K} : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$
  - Defined over finite fields $\mathbb{K}$: $\mathbb{F}_{2^m}$, prime finite field $\mathbb{F}_\mathcal{P}$ or $GF(p)$
  - $\mathbb{F}_\mathcal{P}$ elements for coefficients and coordinates: 200 $\sim$ 400 bits

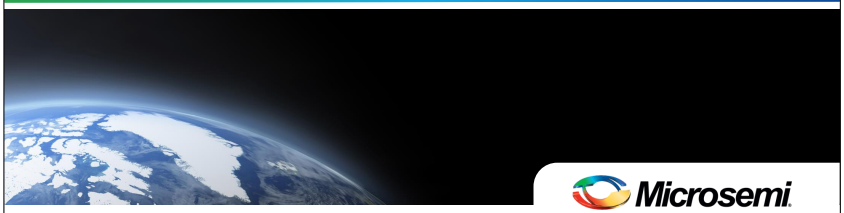
- Hyper-Elliptic Curves
  - **More complex!**
  - Equation $H/\mathbb{K} : y^2 + h(x)y = f(x)$, $deg(h) < g$ and $deg(f) = 2g + 1$
  - g: genus of the curve, $g \leq 2$ in practice for reliable HECC
  - $\mathbb{F}_\mathcal{P}$ elements for coefficients and coordinates: 100 $\sim$ 200 bits

- Kummer surface
  - Not an additive group: no addition law
  - Can be used in HECC using some *(magic)* trick
  - Reduced complexity for curve operations

HTMM Detailed Architecture



Power Matters.



Improving Trust in the FPGA Supply Chain using Blockchain and Keyless-Signature Technology

Richard Newell
Senior Principal Product Architect, SoC Group, Microsemi Corp.
For the Cryptarchi workshop held in
Smolenice, Slovakia June 18-21, 2017

Improving Trust in the FPGA Supply Chain using Blockchain and Keyless-Signature Technology

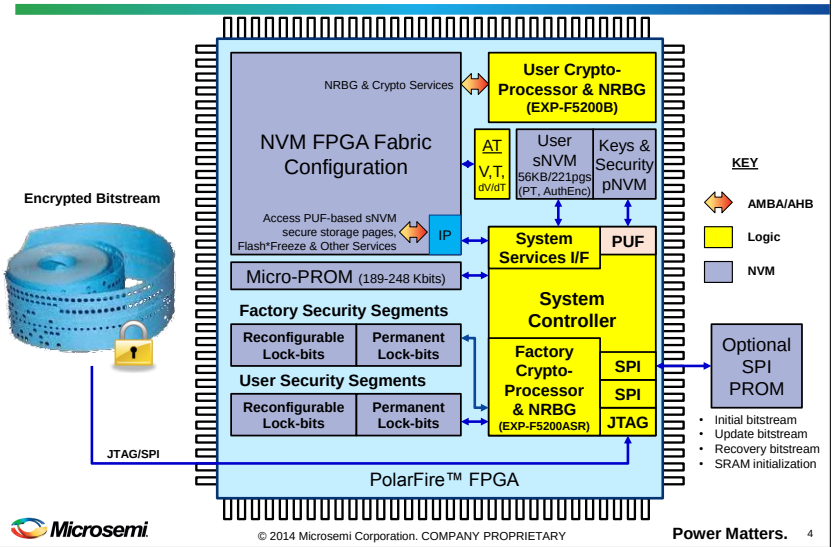
Abstract:

- Blockchain and Keyless Signature Infrastructure (KSI) technologies that only rely on secure message digests and need no secret keys can be used to provide additional assurances that non-volatile FPGA components and systems moving up the supply chain hierarchy from wafer test through to completed systems are trustworthy.
- This is done by providing cryptographic evidence of the FPGA's provenance using a verifiable time-stamped audit trail of key events in the FPGA life-cycle using blockchain and KSI technology, complementing existing traditional measures.
- System manufacturers using those FPGAs can keep appending to the extensible information container that represents the entire history of the FPGA (and eventually the system), strengthening trust in the system, preventing counterfeiting at all levels (component and system), and providing a strong verifiable identity for use in the final run-time application.

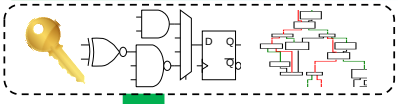
Current Microsemi FPGA Supply Chain Assurance Measures

Implemented and in production for SmartFusion®2,
IGLOO®2 and PolarFire™ FPGAs

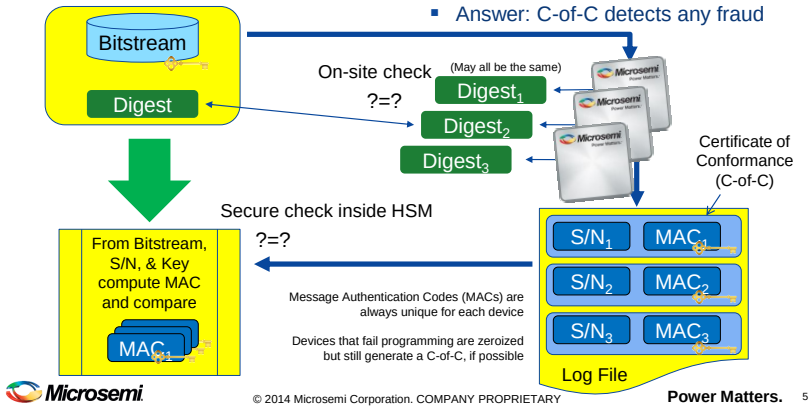
Simplified PolarFire™ FPGA User Security Model



Digital Certificate-of-Conformance (C-of-C)



- How to prevent insiders and rogue CMs from subverting security intent?
  - E.g., loading their own keys, security settings, or a design with a Trojan Horse?
- Answer: C-of-C detects any fraud



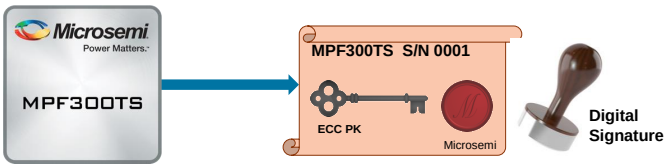
NVM Integrity:
PolarFire FPGA Digest Commands and Services

	JTAG/SPI Command	System Service
Bitstream, IAP, and UIC Authentication Services (external SPI Flash)		X
Export C-of-C tags (during bitstream pgm'g)	X	
Export Digests Stored During Programming (on demand)	X	X
Compute/Export Fresh Digests (on demand)	X	X
Compute/Report Fresh Status Flags (on-demand)	X	X
Compute/Report Fresh Tamper Flag (after Power-on-Reset)		X
Export Zeroization Proof (after zeroization)	X	
Device Integrity Flag (for new devices)	X	
sNVM Authentication (when page is read)		X

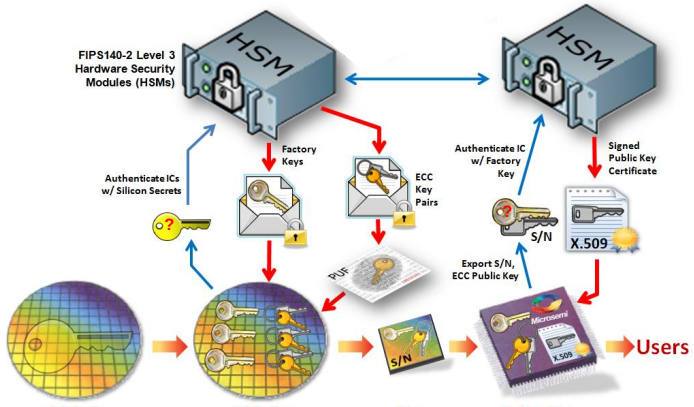
Up to thirteen SHA-256 digests or flags reported, ensuring integrity of the associated NVM

Supply Chain Assurance

- An X.509 conforming certificate digitally signed by Microsemi is stored in each PolarFire device's eNVM
- Certifies integrity and authenticity of contents:
  - Serial number and date code
  - Part Number
  - Includes device's ECC Public Key
- Key verification protocol binds certificate to secret key(s)
- Cryptographically assures customer that each device is…
  - Not counterfeit (nor overbuilt by our fab., nor failed test, etc.)

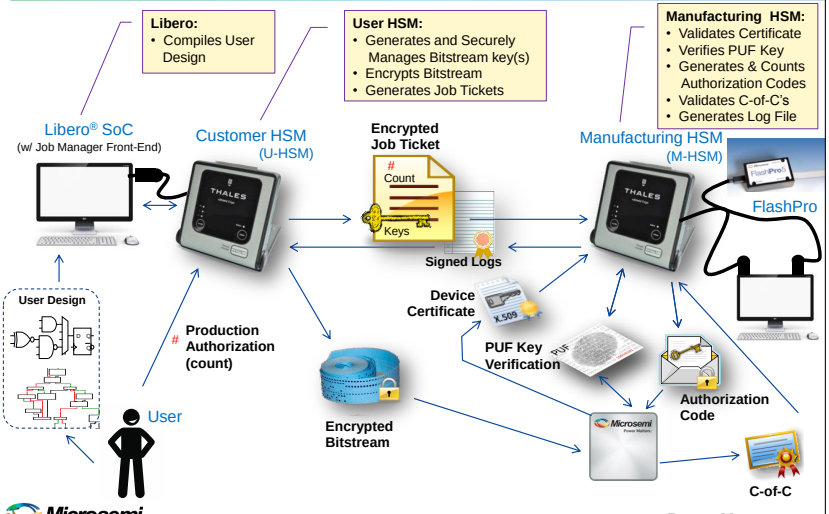


Microsemi HSM-based Manufacturing Flow



SPPS User Device Configuration Data Flow

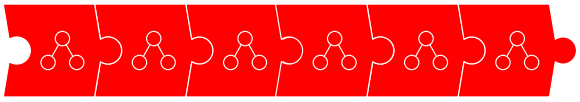
The Secure Production Programming Solution (SPPS)



Guardtime's Keyless Signature Infrastructure (KSI)

Blockchain

Each new block added to the chain includes a message digest (hash) of the previous block



The new block can also contain any kind of digital payload:

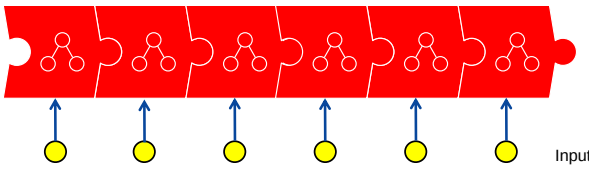
- Ledger entries (e.g., Bitcoin transactions)
- Contracts, documents to record
- Etc.

Public or Private Blockchains? Examples:

- Bitcoin is based on a public blockchain
  - "Proof of Work" entitles someone to add a new block (and earn a reward)
- The Guardtime private blockchain requires prior authorization to submit data to (no proof of work requirement)

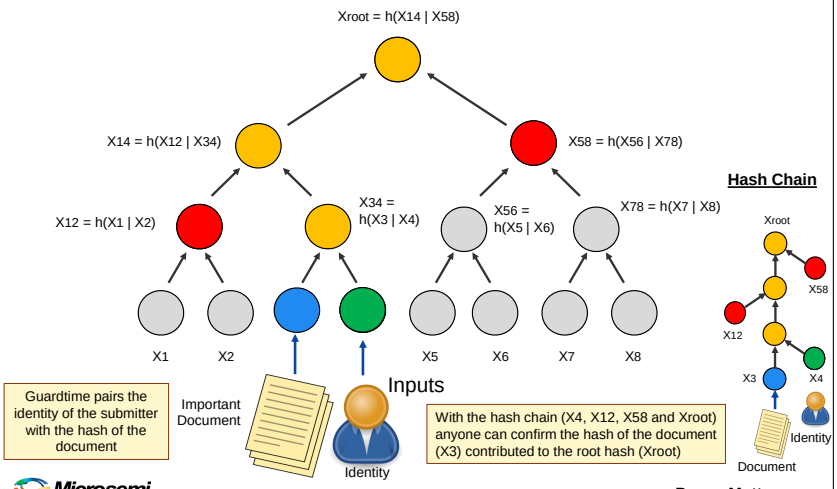
GuardTime's Calendar Blockchain

Adds a block once per second (since "Unix Time" began in 1970)

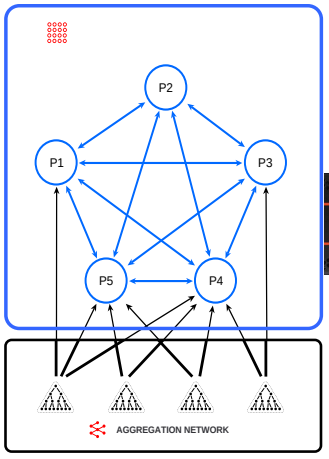


The payload for each added block consists of a single SHA-256 hash value (one per second)
(You will see where this hash comes from shortly)

Merkle Hash Tree Aggregates Inputs



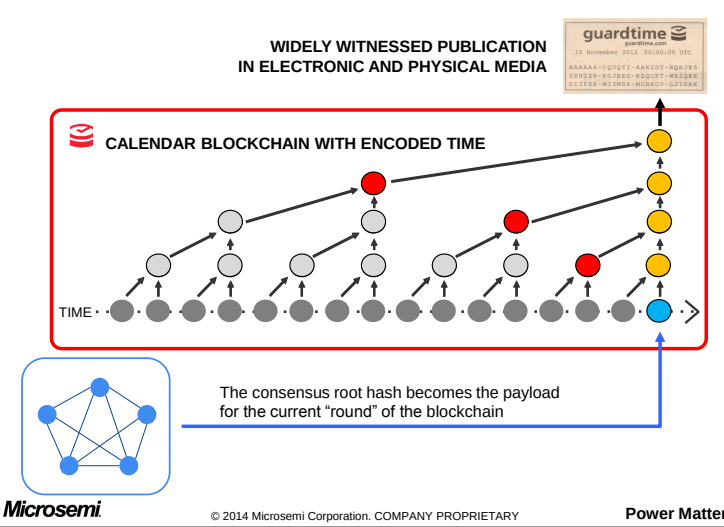
Core Network for Distributed Consensus



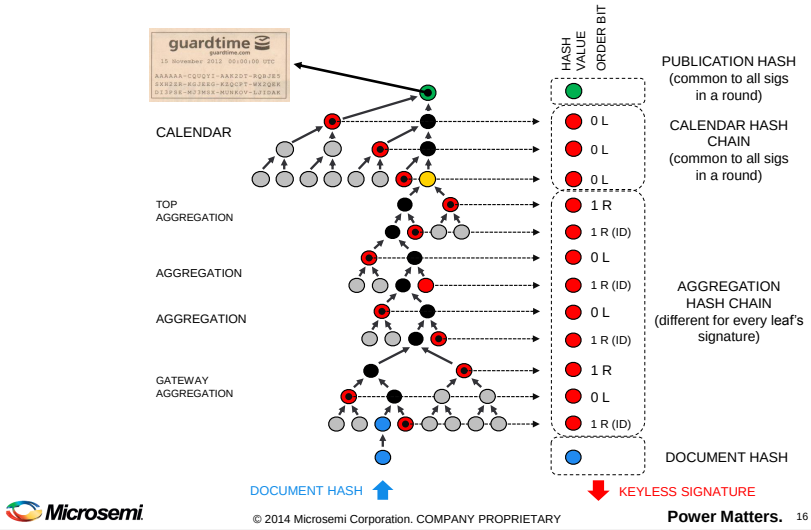
The core network comprises multiple geographically distributed security-hardened servers: Guardtime's Black Lantern™ Security Appliance (HSM)

Core servers receive root hashes from the Aggregation Network (Merkle hash trees) and arrive at a consensus about the hash for each round (second), adding it to the calendar blockchain

Calendar Blockchain w/ Published Trust Anchor



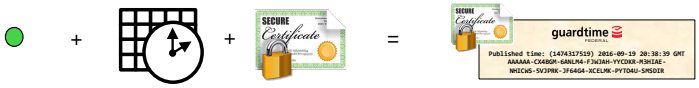
Keyless Signature



KSI Signature Verification
PKI based Trust Anchor

Option 1 – Calendar Authentication Record ("Key-based" verification)

- The publication code in the signature is signed by the core node with a PKI certificate, and embedded in the KSI signature.
- If the PKI Signature is valid,
- And the PKI certificate is trusted, then
- The KSI signature is verified.



PUBLICATION HASH | SIGNING TIME | CORE NODE PRIVATE KEY | CALENDAR AUTHENTICATION RECORD EMBEDDED IN KSI SIGNATURE

KSI Signature Verification
Widely-witnessed publication based Trust Anchor

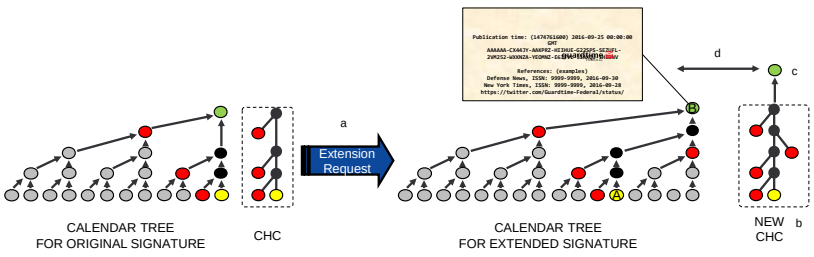
Option 2 – "Extend" the KSI Signature

a. Client requests extension of KSI Signature to later date/time.

b. Extender returns CHC required to get from Aggregation Top Hash (A) at signing time to Publication Hash (B) at later time.

c. Client calculates Publication Hash with Aggregation Top Hash and new CHC.

d. Client compares calculated publication hash to "widely-witnessed" value.



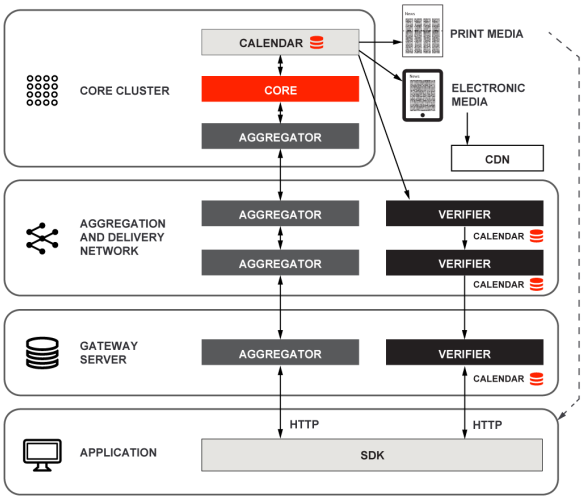
The Keyless Signature Infrastructure

Core Cluster
- Develops consensus and securely adds resulting leaf to the calendar blockchain each second

Aggregation Network
- Provides geographical reach and scale for the system

Gateway Server
- Customer facing server that provides signing and verification services (incl. calendar hash chain extension)

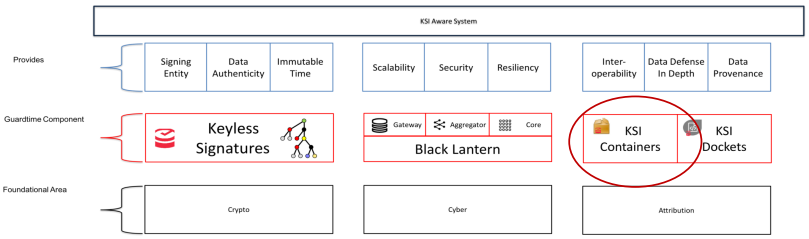
Applications
- KSI-enabled applications request signatures and perform verification.

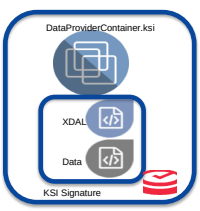


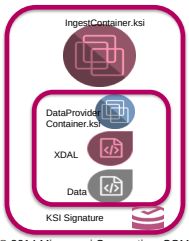
Discussion

- Privacy: KSI does not ingest any customer data; data never leaves the customer premises.
- Scale: KSI performance is practically independent of the number of clients or amount of data signed / verified. Proof of participation is provided in about 2 seconds of request and is not subject to "rollback."
- Open Verification: For KSI Signature verification, one needs to trust publicly available information only - verification does not rely on trusted insiders or security of key-stores.
- Portable: Data can be verified even after crossing geographical or organizational boundaries or service providers.
- Offline: The KSI system does not require network connectivity for verification.
- Supports Real-time Protection: KSI Signature verification requires milliseconds, which allows clients to perform continuous monitoring and tamper detection.
- Long-term validity: Proof is based only on the properties of cryptographic hash functions and does not expire.
- Post-Quantum: The proof stays valid even assuming functioning quantum computers, i.e. does not rely on traditional asymmetric or elliptic curve cryptography.
- Carrier Grade: The KSI system architecture is able to deliver 99.999% availability.

KSI Containers
Extensible Data Attribution Language (XDAL)



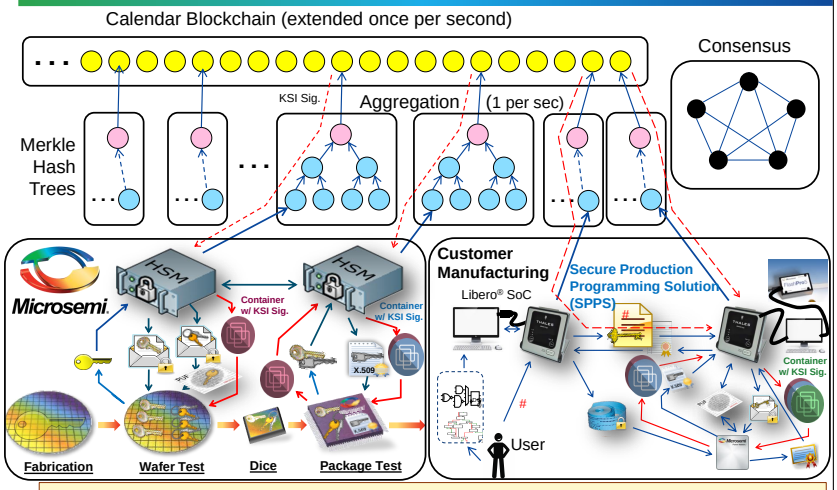




- Containers may wrap any type of data from a hash to a full video
- Containers can ingest other containers (nest-able)

Conceptual FPGA Supply Chain Provenance Application of KSI and XDAL Data Containers

Guardtime Keyless Signature Infrastructure (KSI)
Applied to Microsemi (& User) Trusted Manufacturing Flows



Blockchain records immutable audit trail of key events in device & system lifecycles, supplying provenance

Thank You!

Questions?

Richard Newell
richard.newell@microsemi.com
+1 408 643 6146