Demonstrating an LPPN Processor (Short Paper)

Dina Kamel
ICTEAM/ELEN, Université catholique de Louvain (UCL)

Davide Bellizia
ICTEAM/ELEN, Université catholique de Louvain (UCL)

François-Xavier Standaert
ICTEAM/ELEN, Université catholique de Louvain (UCL)

Denis Flandre
ICTEAM/ELEN, Université catholique de Louvain (UCL)

David Bol
ICTEAM/ELEN, Université catholique de Louvain (UCL)

ABSTRACT

Secure authentication is a necessary feature for the deployment of low-cost IoT devices. Due to their conceptual simplicity, protocols based on the Learning Parity with Noise (LPN) problem have been proposed as promising candidates for this purpose. However, recent research has shown that some implementation issues may limit the practical relevance of such protocols. First, they require a (Pseudo) Random number Generator (RNG) which may be expensive. Second, this RNG may be an easy target for side-channel analysis.

The recently introduced Learning with Physical Noise (LPPN) assumption aims at mitigating these two issues. It removes the need of an RNG by directly performing erroneous computations, which is expected to lead to more efficient implementations and improved side-channel security. So far, the LPPN assumption has only been analyzed mathematically, and its feasibility discussed based on simulations, putting forward the possibility to control the error rate of an implementation thanks to frequency/voltage overscaling.

In this paper, we confirm these promises by demonstrating a first prototype implementation of LPPN in a 28nm FDSOI CMOS technology which occupies an area of 19,400 μm^2. We used a mixed 512-bit parallel/serial architecture in order to limit the exploitation of data-dependent errors with so-called filtering attacks. We additionally designed an on-chip feedback loop that adjusts a variable delay line in order to control the error rate, which prevents other attacks altering external parameters such as the supply voltage, operating temperature and clock frequency. Measurement results show that a simple authentication protocol based on LPPN would consumes 1 μJ per authentication at 0.45V supply. Combined with the excellent algorithmic properties of LPPN regarding security against side-channel and fault attacks, these concrete feasibility results therefore open the way towards the design of full authentication systems with high physical security, at lower cost than standard solutions based on block ciphers.

ASHES'18, October 19, 2018, Toronto, ON, Canada

ACM ISBN 978-1-4503-5996-2/18/10…$15.00
https://doi.org/10.1145/3266444.3266445

CCS CONCEPTS

• **Security and privacy** → **Authentication**; **Hardware Security Implementations**; **Embedded Security**;

KEYWORDS

Authentication; Learning Parity with Noise (LPN); Probabilistic Computations; Side-Channel; Fault Attacks.

ACM Reference Format:
D. Kamel, D. Bellizia, F.-X. Standaert, D. Flandre, D. Bol. 2018. Demonstrating an LPPN Processor . In *The Second Workshop on Attacks and Solutions in Hardware Security (ASHES'18), October 19, 2018, Toronto, ON, Canada.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3266444.3266445

1 INTRODUCTION

The past decade has witnessed the emergence of the Internet of Things (IoT) which allows the integration of embedded devices in an increasingly wide range of applications of the every day life (e.g., smart cars, e-health and smart homes to name a few). This wide spectrum of applications may manipulate confidential data in resource-constrained environments and therefore must ensure minimum security guarantees (e.g., authentication) that can be implemented with a limited area and power/energy consumption budget [24]. The area of lightweight cryptography has been developed in reaction to these new challenges, and a possible solution to solve the lightweight authentication issue is to use block ciphers [7].

Having a lightweight block cipher implementation still ignores a part of the problem though. Indeed, devices deployed in the IoT context are also very accessible targets for physical attacks, where an adversary controlling a cryptographic implementation monitors side-channel leakages [19], or tries to force erroneous computations [13], in order to recover secret data. Unfortunately, state-of-the-art countermeasures against such attacks are usually expensive and difficult to implement – even more if joint countermeasures are needed. For example, the cost of a masked (i.e., secret-shared) implementation which mitigates side-channel leakages roughly grows quadratically in the number of shares [10], and its security depends on physical assumptions that may be contradicted by concrete manufacturing defaults [4, 20]. Furthermore, the interaction between masking and countermeasures against fault attacks may lead to additional overheads [25] and weaknesses [23].

In this context, the Learning Parity with Noise (LPN) problem has gained attention as a versatile and potentially lightweight assumption upon which to build various cryptographic protocols, and in particular authentication ones:

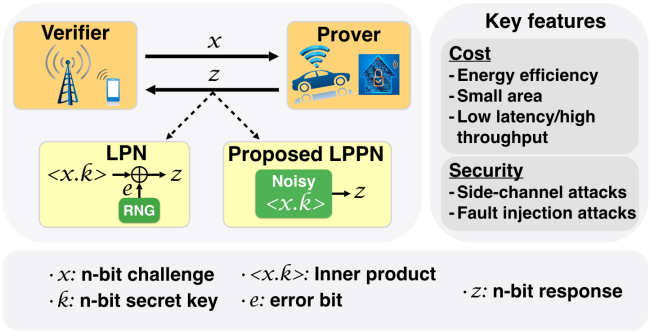


Figure 1: Conceptual block diagram of LPN/LPPN authentication schemes for IoT applications.

see [12] for an early work in this direction, [11] for a more recent proposal and [22] for a survey. The state-of-the-art of these protocols shows a quite opposite trend: while their direct exploitation for lightweight applications is questionable [1], implementations of the inner product computation that is at the core of LPN-based protocols inherently offer good resistance against side-channel attacks via masking (since key-homomorphic) and fault attacks [5]. So they could theoretically lead to masked implementations with overheads that are linear in the number of shares and strong resistance against fault attacks. Yet, two important drawbacks limit the deployment of such solutions: first LPN-based protocols require on-chip (pseudo)randomness generation to produce the noise (which has non-negligible cost); second, this randomness must be protected against physical attacks [8].[1]

In a recent work, Kamel et al. proposed to mitigate these two issues by substituting the Random Number Generator (RNG) needed to implement LPN-based protocols thanks to an inexact implementation of the inner product computations [15]. The latter leads to a new (physical) assumption defined as Learning Parity with Physical Noise (LPPN). As illustrated in Fig. 1, it is appealing since it removes the need of an expensive RNG and prevents side-channel attacks targeting the generation of the noise (since the correct outputs are never explicitly computed by the implementation). Yet, while conceptually elegant, this solution (that was so far only analyzed theoretically and based on simulations) also raises new engineering challenges. In particular, LPPN-based protocols must implement controllers in order to efficiently control the error probability in such a way that an adversary cannot tamper with it externally. In other words, this control should ideally be internal (i.e., on-chip) and autonomous (i.e., automatically adjust the error probability in case of environmental variations provoked by an adversary).

In this work we demonstrate the feasibility of designing a low-power LPPN-based processor using the computations' delays as a control element. Erroneous inner product computations are achieved by sampling their unstable outputs. Concretely, this is achieved by digitally controlling the output sampling clock using a variable delay line. We argue that using a controlled delay line (rather than a controlled clock frequency or supply voltage as suggested in [15]) is interesting since it prevents certain additional attacks. For example, tampering with the clock frequency in order to reduce the error probability and extract the secret key is impossible (since the delay control is independent of the clock frequency). Similarly, the impact of tampering the supply voltage is limited because the variable delay line shares the same supply voltage as the inner product (thus both are impacted in the same way by changes of the supply voltage). Based on this design principle, we describe for the first time a working prototype of low-power 512-bit LPPN processor, fabricated in 28nm FDSOI CMOS technology, where inexact computations are deployed to implement a noisy inner product rather than combining exact inner product computations with a standard RNG to generate the errors. We show how to implement a calibration phase during which the error control is adjusted in order to maintain the required error probability during authentication. We additionally prove that data dependencies in the error probabilities can be kept under control thanks to a mixed (64-bit parallel × 8-bit serial) architecture, as suggested in [15]. We finally validate the correct operation of the LPPN processor (i.e., we verify that the error probability is well within the accepted bounds) across a wide temperature range (-40 °C to 85 °C) – thus eliminating the chances of an adversary who may alter the operating temperature of a working chip in order to try recovering the secret key.

We insist that unlike conventional approximate computing applications where the induced errors are tolerated to improve the performances and save power [3], the goal of the LPPN approximate computations is to rely on the presence of a controllable amount of errors in order to ensure security.

2 BACKGROUND

2.1 LPN problem

Let $k \in \{0,1\}^n$ be a random n-bit secret and $\langle x, k\rangle$ denote the binary inner product of input $x \in \{0,1\}^n$ and k. Let $\epsilon \in]0, \frac{1}{2}[$ be a noise parameter that follows a Bernoulli distribution (Ber_ϵ) such that if $e \leftarrow \mathsf{Ber}_\epsilon$, then $\Pr[e = 1] = \epsilon$ and $\Pr[e = 0] = 1 - \epsilon$, and the distribution $D_{k,\epsilon}$ be defined as:

$$D_{k,\epsilon} =: \{x \leftarrow \{0,1\}^n; e \leftarrow \mathsf{Ber}_\epsilon : (x, \langle x, k\rangle \oplus e)\}.$$

Let $\mathcal{O}_{k,\epsilon}$ denote an oracle outputting independent samples according to the distribution $D_{k,\epsilon}$. The LPN_ϵ^n problem is said to be (q, t, m, θ)-hard to solve if for any algorithm A, the following inequality holds:

$$\Pr[k \leftarrow \{0,1\}^n : A^{\mathcal{O}_{k,\epsilon}}(1^n) = k] \leq \theta,$$

and the algorithm A runs in time $< t$, with memory $< m$ and makes at most q queries to the oracle $\mathcal{O}_{k,\epsilon}$.

2.2 LPN-based authentication

The LPN problem was first used for secret key identification protocols in the proposal by Hopper and Blum [12] (next denoted as the HB scheme) which requires two communication

[1] The latter is not unfeasible but increases the overheads. For example, one solution for masking would be to generate independent Bernoulli noise for all the shares, with probabilities of errors per share set as a function of the number of shares thanks to the piling-up lemma.

rounds. Later, it was extended to HB$^+$ [14] that is secure in an active attack model, and to HB$^\#$ that is secure against man-in-the-middle attacks [9]: both require three communication rounds. In [16] a two-round alternative identification scheme was proposed, which was the basis for the LAPIN protocol [11]. The LPPN processor that we study in this paper can be used for the implementation of all these protocols.

2.3 LPPN problem

To describe the LPPN assumption, we first recall the "physical function" defined in [2], slightly simplified as in [15]. A physical function $\mathsf{PF}_{d,\alpha}$ is a probabilistic procedure based on a physical device d, which can be stimulated with some input challenge $x \in \{0,1\}^{n_i}$, making d respond with a (probabilistic) output $y \in \{0,1\}^{n_o}$, with α a set of parameters and n_i and n_o the number of input and output bits. In our "physical LPN" context, d is the implementation of an approximate inner product computation, so $n_i = n$ and $n_o = 1$.

A physical function is called an $\tilde{\epsilon}$-Physical Inner Product ($\tilde{\epsilon}$-PIP) if, on uniform public input $x \in \{0,1\}^n$ given $k \in \{0,1\}^n$ be a random n-bit secret stored in the device d_k, it outputs $\langle x,k\rangle$ with estimated error probability: $\hat{\Pr}[\mathsf{PF}_{d_k,\alpha}(x) \neq \langle x,k\rangle] = \tilde{\epsilon}$. Based on this definition, the LPPN problem can be described just as the LPN problem, with the only difference that the LPN samples are replaced by the outputs of an $\tilde{\epsilon}$-PIP. It is conjectured in [15] that the LPPN problem can be as hard as the LPN problem.

3 PIP IMPLEMENTATION

In summary, the main technical challenge to leverage the LPPN assumption in concrete applications is to design an $\tilde{\epsilon}$-PIP such that the error probability $\tilde{\epsilon}$ cannot be tampered with by an adversary manipulating the implementation generating the noisy samples. Figure 2 shows the high-level architecture of a LPPN processor aimed to fulfill this requirement. It consists of an inner product block, a variable delay line and a finite state machine (FSM) error controller. The operation of this processor holds in two phases: first, a calibration phase where the verifier exchanges sets of challenge-response (q queries) during r steps with the prover to tune its control (so that it responds with the required error probability during the authentication); second the actual authentication.

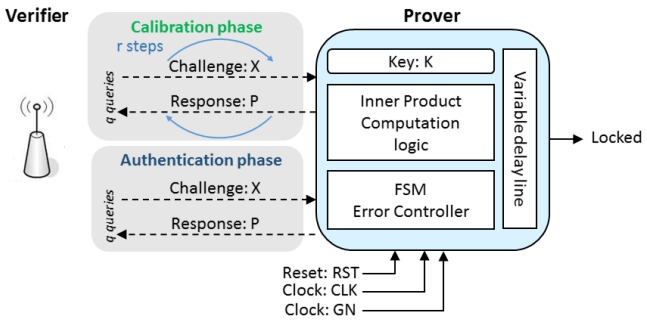


Figure 2: LPPN processor high-level architecture.

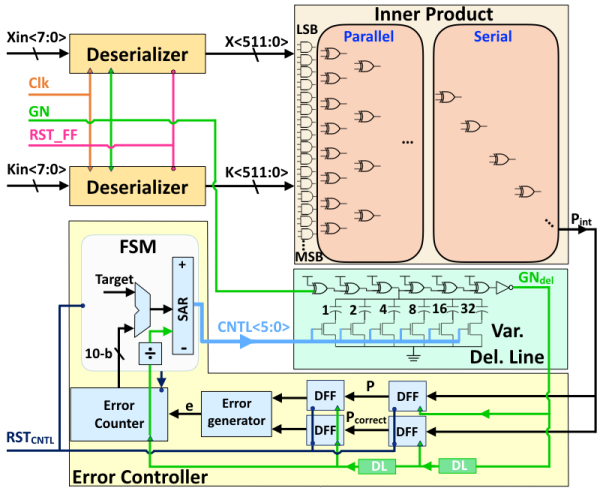


Figure 3: LPPN processor block diagram.

The LPPN processor is detailed in Fig. 3. It also implements a deserializer stage to facilitate the communication of 512-bit values to the chip. In our current prototype implementation (of which the goal is to investigate the error control part of an $\tilde{\epsilon}$-PIP), both the key and the input are sent to the LPPN processor. In a real-world prototype, the secret key would be embedded (in a shared manner if masked) and the random input would also be generated on-chip in case of protocols secure against man-in-the-middle attacks. The architecture of the inner product block, which computes the parity signal P_{int}, is a $(p \times s)$-bit mixed architecture, where p is a 64-bit parallel stage and s is an 8-bit serial stage. The choice of this architecture is justified by the fact that it reduces the possibility that an adversary exploits a data-dependent error probability thanks to so-called filtering attacks, as discussed in [15] and experimentally validated next. The variable delay line uses digitally-controlled delay elements with shunt-capacitors via NMOS switches. To control these switches, an error controller provides a 6-bit control signal (CNTL) which is adjusted based on the required probability during the calibration phase. It comprises a pair of DFFs to output an erroneous parity P and a correct (delayed) version of it, $P_{correct}$, an error generator to compare both bits and compute the error signal e, a 10-bit error counter to count the number of errors over 1024 queries and a comparator that compares the error count to the target (e.g., 256 to achieve $\epsilon = 0.25$). The $CNTL$ bits are then set to 1 or 0 through the FSM in a successive approximation scheme.

The timing diagram in Fig. 4a shows both the key (K_{in}) and challenge (X_{in}) that are shifted through the deserializer's flipflops during 64 clock cycles and then loaded via the latches during the low state of the GN clock signal. The pulsed GN clock signal is delayed through the variable delay line stages and inverted (it is now renamed GN_{del}) in order to sample the output of the inner product block P_{int} during its glitchy period. The erroneous output parity bit P is then compared to the correct parity bit $P_{correct}$, thus generating the error signal e. It is important to emphasize that the $P_{correct}$ bit

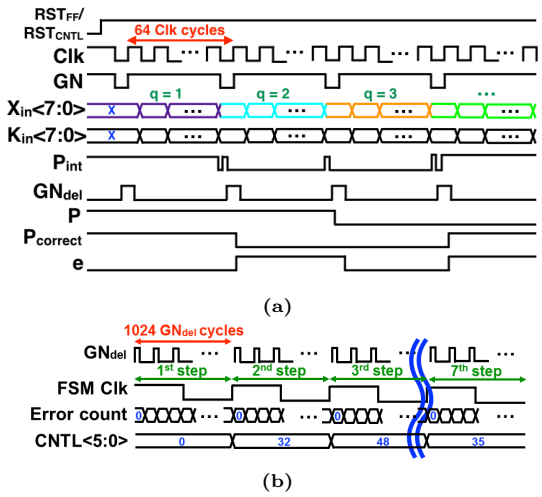


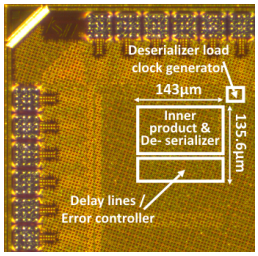
Figure 4: Timing diagram of the LPPN processor (part a) and of its error control block (part b).

is only needed during the calibration phase and should not be available during the actual authentication phase, otherwise the probing-like attacks that the LPPN assumption is supposed to mitigate become trivially applicable again.

The timing diagram of the error control block is shown in Fig. 4b. The errors are counted in the control block during one FSM clock cycle ($T_{FSM} = 1024 \times T_{GN_{del}}$). During the first step, the 6 $CNTL$ bits are all reset to zero (i.e., the delay of the variable delay line is at its minimum which implies a high error count). At the end of the first step, the 10-bit error count is compared to the target count and the FSM decides to set the MSB of the $CNTL$ to one during the second step in order to increase the delay of the GN_{del} edge. The following steps are conducted in the same way where the FSM sets the corresponding $CNTL$ bit to one and decides whether to reset the previous bit to zero or not until all bits of the $CNTL$ signal are computed. This in total requires 7 steps (1 step where all bits are reset to zeros and 6 steps to compute the values of the 6 $CNTL$ bits). After the $CNTL$ bits are computed, the error controller goes into lock state, keeps the same values of the $CNTL$ bits as in the last step, and the actual authentication can take place.

4 MEASUREMENT RESULTS

The LPPN processor including the deserializer, the delay line and the error controller circuits occupies 19,400 μm^2 in a 28nm FD-SOI process using CMOS logic gates as shown in Fig. 5. The full system consumes 20.16 μW from a 0.45 V supply. The LPPN processor is clocked by an external clock of 10 MHz provided by National Instruments' digital waveform generator (PXI-6552) which also generates the 8-bit serial input X_{in} and key K_{in} sequences. The chip also features the deserializers' load clock GN generator. It operates in two modes: the normal mode where the error controller searches for the appropriate $CNTL$ bits to achieve the required error probability and a testing mode, where the $CNTL$ bits are



number of bits (n)	512
number of queries (q)	1024
Target Pr[e = 1] (ϵ)	0.25
Upper bound (ϵ_{max})	0.348
Lower bound (ϵ_{min})	0.125

Figure 5: Die Micrograph of the LPPN processor and typical parameters for an authentication scheme.

externally generated so that the error probability can be swept across the full $CNTL$ range. The system parameters are set as in Fig. 5 (from [17]). The maximum boundary of ϵ is 0.348 which mitigates the probability to reject a honest prover, while the minimum boundary (0.125) ensures the security of LPPN does not fall below 82 bits [15]. As illustrated in Fig. 6, the chip behaves as expected with internal automatic $CNTL$ calibration, and the final error probability = 0.225 which is very close to the target $\epsilon = 0.25$ and well within the boundaries tolerated by the LPN authentication scheme.

4.1 Data dependencies

The security of LPPN-based protocols requires the error probability to be independent of the inputs. Otherwise an adversary can exploit correlations between the inputs and outputs of the approximate computation of an $\tilde{\epsilon}$-PIP by filtering the input challenges (in order to reduce the error probability). In order to mimic this scenario, we set some m-LSBs of the secret key to 0. Since the LPPN assumption only works with random inputs, setting bits to 0 requires that the challenges are filtered (which has an exponential cost and is typically unfeasible for $m > 64$). Figure 7 demonstrates the measurement results of the mixed LPPN system where the control signal is externally swept from 0 to 60 and the error probability is computed over 1024 queries, first with a random fixed key, then using keys whose m-LSBs are set to 0 (m takes values from 8 to 256). It is clear that the deviations of the error probability due to setting the key's LSBs to 0 is sufficiently limited up to 64 bits and would be tolerated

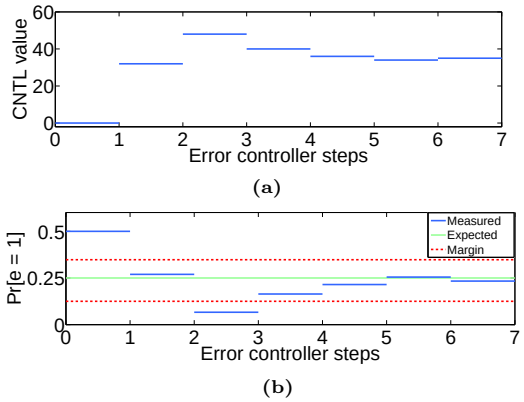


Figure 6: (a) Error measured by the LPPN controller in 7 successive steps thanks to the CNTL signal and (b) the corresponding error probability.

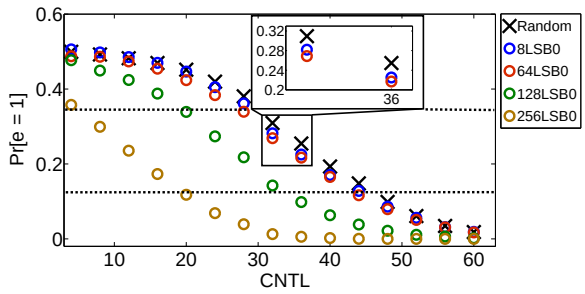


Figure 7: Impact of filtered challenges on the error probability of the mixed PIP architecture.

within the boundaries of the LPPN authentication scheme. Therefore, we conclude that the data dependencies (due to deterministic effects) in our architecture can be kept small enough to ensure security against filtering attacks.

4.2 Voltage-Temperature sensitivity

In order to maintain the error probability well within bounds, the LPPN system needs to adapt autonomously to any change imposed by an adversary such as altering the system clock frequency, the chip's voltage supply or simply the temperature. Since the error probability is controlled by adjusting the delay of the output sampling clock edge, it is naturally independent of the system clock frequency. Fig. 8a shows the impact of varying the supply voltage from 0.4 V to 1 V and the temperature from -40 °C to 85 °C (using a climate chamber) on the $CNTL$ signal that reaches its maximum value (63) at a supply of 1 V and at 85 °C. The corresponding error probability across this range is well within the defined bounds: $(\epsilon_{min} = 0.125) <$ error probability $< (\epsilon_{max} = 0.348)$, as presented in Fig. 8b. The white region in the Shmoo plots represents the conditions where the chip does not fulfill its security conditions anymore (when runinng at 10 MHz).[2]

4.3 Performance assessment

The performance summary of the LPPN processor for an exemplary authentication is given in Table 1. The reported area is for the full implementation. The overhead of the control part (i.e., the delay lines and the error control) count for 10 % (in terms of number of instances) while the flip-flops,

[2] In the above experiment, we assume an adversary who tampers with either the supply voltage or the temperature (or both) in the same way during the calibration and the actual authentication phases. In such a case, whatever change he enforces on the system is compensated by the control block such that the error probability is well bounded within the specified range. A more dangerous scenario could occur if the adversary is capable of altering the supply voltage and/or temperature during the calibration phase in a different way than during the authentication phase. In this case, the $CNTL$ signal locked during calibration will no longer correspond to the conditions of the authentication. As a result, the error probability could be reduced and the secret key recovered. One way to overcome this problem is to implement an on-chip voltage regulator that has good line regulation (e.g. [21]). Another solution is to implement a voltage sensing mechanism such that if the voltage is different between the calibration and the authentication phases, the authentication is canceled. Designing such protective schemes is a scope for future research. Note that such attacks would be much harder with the temperature, because the latter changes between hugely different values (see Fig. 8a) and an attack would therefore need more time than the calibration time of the LPPN processor (less than 50 ms).

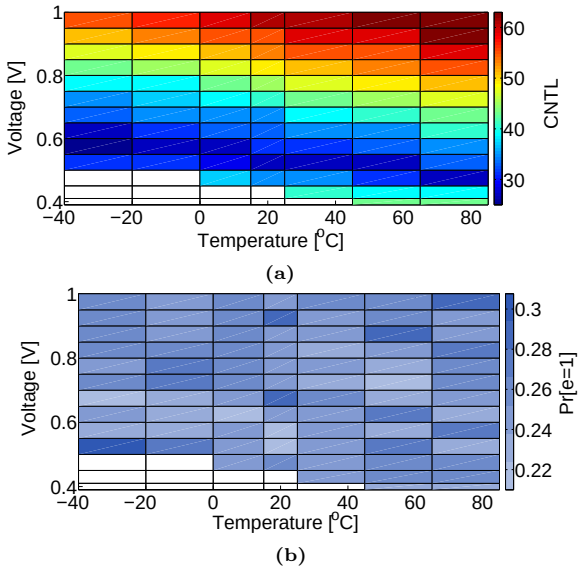


Figure 8: Impact of voltage supply and temperature on (a) the error controller's output (CNTL) and (b) the probability of error after the calibration.

latches of the deserializers and the inner product block are the dominant contributors. The overall energy per authentication is 1 μJ at 0.45 V supply and corresponds to a total of 8 steps (i.e., 7 steps to calibrate the error control signal and one for the actual authentication). This assumes that calibration is repeated for each authentication in order to avoid attacks controlling the supply voltage or temperature. As mentioned in Footnote 2, using on-chip voltage regulators would allow to mitigate such attacks (and therefore to perform calibration only once, or at least much less regularly, potentially reducing the energy cost by a factor close to 8 and reducing the communication complexity – which is an interesting research direction). The LPPN processor consumes a total of 20 μW power at 0.45 V. The latency which is the time for a full authentication cycle (including the $CNTL$ calibration phase) is 52 ms at 10 MHz clock frequency. The throughput of the LPPN processor is reported to be 156.25 kbps.

5 DISCUSSION & FUTURE WORK

As discussed in [15], the relevance of the LPPN assumption may come from different sources of hardness. The minimum requirement (investigated in the previous sections) is that the error probability is set to a value similar to those used by the LPN assumption and cannot be tampered with by an adversary. More precisely, this is sufficient in the case

Table 1: LPPN processor summary.

Area	19,400 μm^2 16,333 GE
Energy per HB authentication	1 μJ
Power @ 0.45V	20.16 μW
Latency	52 ms
Throughput	156.25 kbps

of a masked implementation of LPPN, which is its typical use case: see [15], Section 9.2. But theoretically, the security of the LPPN assumption could also be obtained directly thanks to deterministic and probabilistic physical effects (see [15], Section 8.2). In order to analyze these effects, we superimposed a noise signal over the voltage supply using the AG33250A Keysight arbitrary waveform generator. In this experiment, 1024 random challenges and a random fixed key are repeated 100 times at low ($\sigma_n = 4.6$ mV) and high ($\sigma_n = 332$ mV) supply voltage noise. To distinguish the deterministic and probabilistic components, we observe the output parity bits over the repeated input challenges. For the 1024 output parity bits, we compute the number of repetitions over the 100 times. If the output party bit is repeated more than 95 times it is considered deterministic. The deterministic component is the percentage of parity bits in the 1024 pattern that satisfies this condition. The probabilistic component is 100 % minus this value. In Fig. 9 we demonstrate how supply noise changes the contribution of each component at two different supply voltages (0.45 V and 0.6 V). It is clear that the deterministic component of the LPPN processor is dominant at 0.6 V and low supply noise (91.7 %), and the contrary holds at higher supply noise when the supply voltage is reduced to 0.45 V. The deterministic component never reaches 100 % because of jitter on the clock signal (which was measured to be 34 ps). Similarly, the probabilistic component never reaches 100 % in our experiment because of the limitation of the measurement setup where we reached the maximum supply noise.

We tested the deterministic Boolean function corresponding to the upper left case of the figure and observed the following cryptographic properties [6]: algebraic degree 7, algebraic immunity 3, fast algebraic immunity 6, non-linearity 68 (leading to a bias of ≈ 0.25) and most critically unbalanced (only 106 ones in its truth table). In short, it implies that this Boolean function (despite not being trivially weak from the non-linearity/algebraic viewpoints) is not sufficient to be used as a filter that would lead to secure LPPN based only on deterministic effects. Furthermore, its unbalanced nature means that as long as a purely probabilistic $\tilde{\epsilon}$-PIP is not reached, there will remain exploitable correlations (in our case where the errors originate mainly from an 8-bit serial part).

These last observations are not problematic for the relevance of the LPPN assumption. They recall that building

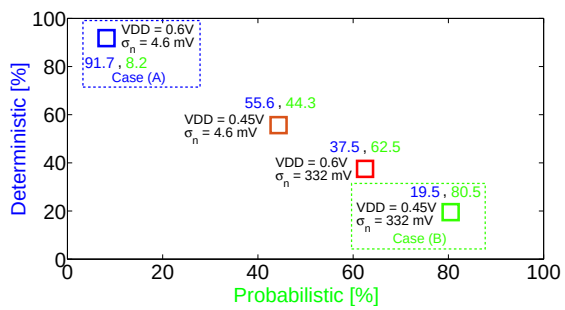


Figure 9: Impact of supply noise and voltage on the randomness of the LPPN processor output.

cryptographic primitives based on physical effects is challenging (and hard to assess), as witnessed by the Physically Unclonable Functions (PUF) literature [18]. In this respect, it is interesting that the LPPN assumption can lead to secure protocols even without deterministic or probabilistic physical hardness (i.e., controlling the error is enough in the case of masked LPPN). LPPN can use the physics to make physical (side-channel and fault) attacks harder and rely on mathematical hardness for black box security. Based on these results, the design of a full LPPN system masked against side-channel analysis and potentially using the improvements mentioned in Footnote 2 are promising directions for future work.

Acknowledgments. Work funded in parts by the ARC Project NANOSEC and the ERC project SWORD.

REFERENCES

[1] F. Armknecht et al. Lightweight authentication on ultra-constrained rfids - myths and facts. In *RFIDSec*, pg 1–18, 2014.
[2] F. Armknecht et al. A formalization of the security features of physical functions. In *IEEE S&P*, pg 397–412, 2011.
[3] L. Avinash et al. Synthesizing parsimonious inexact circuits through probabilistic design techniques. *ACM Trans. Embedded Comput. Syst.*, 12(2s):93:1–93:26, 2013.
[4] J. Balasch et al. On the cost of lazy engineering for masked software implementations. In *CARDIS*, pg 64–81, 2014.
[5] F. Berti and F. Standaert. An analysis of the LPN assumption against fault attacks. In *CARDIS*, pg 245–264, 2016.
[6] C. Carlet. *Boolean Models and Methods in Mathematics, Computer Science, and Engineering*, chapter Boolean Functions for Cryptography and Error Correcting Codes., pg 257–397. 2010.
[7] T. Eisenbarth et al. A survey of lightweight-cryptography. *IEEE Design & Test of Computers*, 24(6):522–533, 2007.
[8] L. Gaspar et al. Hardware implementation and side-channel analysis of LAPIN. In *CT-RSA*, pg 206–226, 2014.
[9] H. Gilbert et al. HB$^{\#}$: Increasing the security and efficiency of HB^{+}. In *EUROCRYPT*, pg 361–378, 2008.
[10] V. Grosso et al. Masking vs. MPC: how large is the gap for the AES? *J. Cryptographic Engineering*, 4(1):47–57, 2014.
[11] S. Heyse et al. Lapin: An efficient authentication protocol based on ring-LPN. In *FSE*, pages 346–365, 2012.
[12] N. J. Hopper and M. Blum. Secure human identification protocols. In *ASIACRYPT*, pg 52–66, 2001.
[13] M. Joye and M. Tunstall, editors. *Fault Analysis in Cryptography*. Information Security and Cryptography. Springer, 2012.
[14] A. Juels and S. A. Weis. Authenticating pervasive devices with human protocols. In *CRYPTO*, pg 293–308, 2005.
[15] D. Kamel et al. Learning with physical noise or errors. *IEEE Trans. on Dep. and Secure Computing (TDSC)*, pg 1–1, 2018.
[16] E. Kiltz et al. Efficient authentication from hard learning problems. *J. Cryptology*, 30(4):1238–1275, 2017.
[17] É. Levieil and P. Fouque. An improved LPN algorithm. In *SCN*, pg 348–359, 2006.
[18] R. Maes and I. Verbauwhede. Physically unclonable functions: A study on the state of the art and future research directions. In *Towards Hardware-Intrinsic Security* , pg 3–37. 2010.
[19] S. Mangard et al. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007.
[20] S. Nikova et al. Secure HW implementation of nonlinear functions in the presence of glitches. *J. Cryptology*, 24(2):292–321, 2011.
[21] S. Peng et al. A power-efficient reconfigurable output-capacitor-less low-drop-out regulator for low-power analog sensing front-end. *IEEE TCAS*, 64-I(6):1318–1327, 2017.
[22] K. Pietrzak. Cryptography from learning parity with noise. In *SOFSEM*, pg 99–114, 2012.
[23] F. Regazzoni et al. Interaction between fault attack countermeasures and the resistance against power analysis attacks. In Joye and Tunstall [13], pg 257–272.
[24] A. Sadeghi et al. Security and privacy challenges in industrial internet of things. In *DAC*, pg 54:1–54:6, 2015.
[25] T. Schneider et al. ParTI - towards comb. countermeasures against side-channel & fault attacks. In *CRYPTO* pg 302-332, 2016.