Oblivious Transfer based on Physical Unclonable Functions (Extended Abstract)

Ulrich Rührmair

Computer Science Department
Technische Universität München
85748 Garching, Germany
ruehrmai@in.tum.de
http://www.pcp.in.tum.de

Abstract. Oblivious transfer (OT) is a simple, but powerful cryptographic primitive, on the basis of which secure two-party computation and several other cryptographic protocols can be realized. In this paper, we show how OT can be implemented by Strong Physical Unclonable Functions (PUFs). Special attention is thereby devoted to a recent subclass of Strong PUFs known as SHIC PUFs. Our results show that the cryptographic potential of these PUFs is perhaps surprisingly large, and goes beyond the usual identification and key exchange protocols.

1 Introduction

Motivation and Background. Electronic devices are becoming increasingly mobile, cross-linked and pervasive, which makes them a well-accessible target for adversaries. Mathematical cryptography offers several measures against the resulting security and privacy problems, but they all rest on the concept of a secret binary key: They presuppose that the devices can contain a piece of information that is, and remains, unknown to an adversary. This requirement can be difficult to uphold in practice: Invasive, semi-invasive, or side-channel attacks, as well as various software attacks including viruses, can lead to key exposure and full security breaks.

The described situation was one motivation that led to the development of *Physical Unclonable Functions (PUFs)* [1]. A PUF is a (partly) disordered physical system S that can be challenged with so-called external stimuli or challenges C_i, upon which it reacts with corresponding responses R_i. Contrary to standard digital systems, a PUF's responses shall depend on the nanoscale structural disorder present in it. It is assumed that this disorder cannot be cloned or reproduced exactly, not even by the PUF's original manufacturer, and that it is unique to each PUF.

Due to their complex internal structure, PUFs can often avoid some of the shortcomings associated with digital keys. It is usually harder to read out, predict, or derive their responses than to obtain the values of digital keys stored in non-volatile memory. This fact has been exploited for various PUF-based security protocols, for example schemes for identification [1] and key exchange [2].

Oblivious Transfer. Oblivious transfer (OT) is a two-player cryptographic primitive which was originally introduced by [3] [4]. Several variants exist, which are reducible to each other [5] [31]. The version considered in this paper is a one-out-of-two oblivious transfer or $\binom{2}{1}$-OT [5]. This is a protocol with the following properties: At the beginning of the protocol, one party Alice (the "sender") holds two secret bits b_0 and b_1 as private input, and another party Bob (the "receiver") holds a secret choice bit c as private input. After execution of the protocol, the following conditions must be met: (i) Bob has learned the bit b_c, i.e. those of the two bits b_0 and b_1 that was selected by his choice bit c. (ii) Even an actively cheating Bob cannot derive any information about the other bit $b_{c\oplus 1}$ as long as Alice follows the protocol. (iii) Even an actively cheating Alice cannot learn c if Bob follows the protocol.

Since its introduction, a large class of cryptographic schemes has been realized on the basis of OT, including bit-commitment, zero-knowledge proofs, and general secure multi-party computation [6] [7] [8] [9] [10]. This makes OT a very versatile and universal primitive. The fact that OT can be realized within a certain cryptographic model is often seen as an indication of the model's large cryptographic potential. For these reasons, the feasibility of OT in the context of quantum cryptography [11] [12], within the Bounded Storage Model (BSM) [14] [15], or in noise-based cryptography [16] [17], has been well-investigated in earlier publications .

Our Contribution. In this extended abstract, we describe a protocol that implements oblivious transfer on the basis of two types of Physical Unclonable Functions: So-called Strong PUFs and SHIC PUFs (see Section 2). The protocol seems to indicate the large potential of these PUFs beyond the known schemes for identification [1] and key exchange [2].

The protocol can be executed between two players Alice and Bob under the following prerequisites: (i) Bob had previous access to the Strong PUF/SHIC PUF in a pre-setting phase. During this phase, he established a list of challenge-response-pairs (CRPs) of the PUF, which is unknown to Alice. (ii) At the time of protocol execution, the Strong PUF/SHIC PUF has been transfered to Alice. Only Alice has access to it and can measure CRPs of the PUF.

Since it is known from other publications that OT is a symmetric primitive [31], our technique allows OT in both directions under the above provisions, without re-transferring the PUF from Alice to Bob (see Sec. 3.3).

Organization of the Paper. In Section 2, we give some background on the two specific PUF types which are relevant for this paper (i.e., Strong PUFs and SHIC PUFs). We also briefly discuss their implementation. In Section 3 we describe and analyze our protocol for oblivious transfer. We conclude the paper in Section 4.

2 Background on PUFs

We now give some background on the two PUF types relevant for this paper. Since SHIC PUFs are a special form of Strong PUFs, we start with an explanation of the latter.

2.1 Strong PUFs

A Strong PUF [1] [18] is a (partly) disordered physical system S, which can be excited with a finite number of external stimuli or challenges C_i, upon which it reacts with corresponding responses R_{C_i} [2]. The pairs (C_i, R_{C_i}) are usually called the challenge-response pairs (CRPs) of the PUF. Three security relevant properties of a Strong PUF S are the following:

(i) Due to the disordered microstructure of S, it must be practically infeasible to fabricate a physical clone S' of S, which has the same challenge-response behavior as S. This restriction shall even hold for the original manufacturer of S.
(ii) Due to the large number of possible challenges that can be applied to S, it must be practically infeasible to exhaustively measure all CRPs of S within a limited time frame on the order of weeks, months, or even years.
(iii) Due to the complicated internal interactions of S, it must be practically infeasible to devise a computer program that correctly predicts the response of S to a randomly chosen, previously unknown challenge with high probability. This should hold even if many other challenge-response pairs of S are known.

Together, conditions (i) to (iii) imply that the responses R_{C_i} of S can be determined correctly (with high probability) only by someone who has got direct physical access to the single, unique PUF S. Implementation examples of Strong PUFs include complex optical scatterers [1] or integrated circuits whose outputs depend on their internal, individual runtimes delays [21] [22] [23]. Also analog cellular arrays have been proposed recently [24].

It has been realized relatively early, however, that machine learning techniques are a natural and powerful tool that can potentially challenge the above security condition (iii). Successful attacks on several Strong PUF candidates have indeed been reported in [20] [21] [25] [26]. To rule out a potential susceptibility to algorithmic modeling attacks, SHIC PUFs have been introduced.

2.2 SHIC PUFs

SHIC PUFs are pronounced as "chique PUFs" and have been suggested in [27] [28] [29]. The acronym "SHIC" stands for Super High Information Content. They are Strong PUF (i.e. they possess the above properties (i) to (iii)) and have the following additional features:

(iv) They contain an extraordinarily high amount of response-relevant random information and have a very high information density.

[1] Strong PUFs also have been referred to simply as PUFs [19], as Physical Random Functions [19] [21] [22] , or, almost equivalently, as Physical One-Way Functions [1].

[2] Please note that the terminology "R_{C_i}" slightly deviates from the standard terminology "R_i" for PUFs. The new terminology is introduced here in order to make the description of Protocol 2 less ambiguous.

(v) Their read-out speed (i.e. the frequency by which they produce responses) is limited to low values. This limitation should not be enforced by an artificially slow access module or the like, which could potentially be circumvented or cut off by suitable invasive means. Rather, it must be an inherent property of the PUF's design and its physical properties.
(vi) The CRPs of a SHIC PUF are mutually independent. The pairwise mutual information of any two responses of theirs is zero.

SHIC PUFs can be imagined as a huge read-only memory with a very high random information content and an *intrinsically slow read-out speed*. A challenge C_i to a SHIC PUF is the analogue to the address in a classical memory, and the corresponding response R_{C_i} is similar to the bit-value stored under that address. A possible realization with concrete numbers for information content, information density and read-out speed will be discussed in Section 2.3.

Strong PUFs vs. SHIC PUFs. As emphasized earlier, all SHIC PUFs are Strong PUFs, but they possess the further properties (iv) to (vi) above. SHIC PUFs thus have the advantage that their security does not depend on the computational power and the machine learning capabilities of the attacker. As all their CRPs are independent of each other, they withstand prediction even by attackers with unlimited computational power until a complete read-out has been accomplished. Their security only depends on the CRPs known to an adversary vs. the overall number of CRPs of the PUF.

2.3 Realization of SHIC PUFs

Even though this is not the main topic of this manuscript, we will briefly discuss the practical realization of the theoretical concept of a SHIC PUF. One potential candidate are ALILE-based Crossbar PUFs, which have been introduced in [27] [28] [29]. We will only provide a short overview of this approach; much further detail can be found in [27] [28] [29].

Generating Randomness by the ALILE Process. Any SHIC PUF must contain a very large random information content. There are many physical processes that generate large entropy in solid-state systems, but one example that can eventually lead to integrated electrical realizations of SHIC PUFs is a process known as ALuminum-Induded Layer Exchange (ALILE) [27] [28] [29]. It is a simple, crystallization-based method that employs only inexpensive starting materials (amorphous silicon and aluminum). It result in polycrystalline films with p-type conduction, which exhibit a very large level of disorder and randomness (see Fig. 1 a). By adjusting the process parameters, the size, number and density of the crystallites can be tuned as desired. The randomness causes individual electrical properties in different subregions of the surface.

Crossbar-based Read-Out. One method that was investigated in [27] [28] [29] is to read out the information from ALILE structures by so-called crossbar architectures. Slightly simplifying, a crossbar consists of two sets of parallel wires, which are attached to the top and to the bottom of the crystallized structure. The bottom set of wires

is arranged in a 90° angle to the top set, as shown in Figure 1. If source and drain voltages are applied at exactly one top and one bottom wire, current flows through the polycrystalline film area at the virtual crossing of the two wires. $I(V)$ curves with a strongly rectifying behavior [29] are observed, which depend on the individual, random configuration in the polycrystalline substrate at the crossing. They can be converted into a few bits of individual information per crossing [27] [28].

Crossbar architectures are among the simplest functional nano devices and possess a very regular geometry. They can hence be fabricated with very small inter-wire distances, leading to high information densities. Concrete realization parameters we tried to make plausible by measurement on single diodes and by crossbar simulations in [28] are 10^5 top wires and 10^5 bottom wires, which leads to an information of around 10^{10} bits per cm^2. This assumes that the footprint of one crossing is 100 nm $\times$ 100 nm [28]. A single CRP of such a structure would have a length of around around $1 + 2 \cdot \log_2 10^5 \approx 35$ bits.

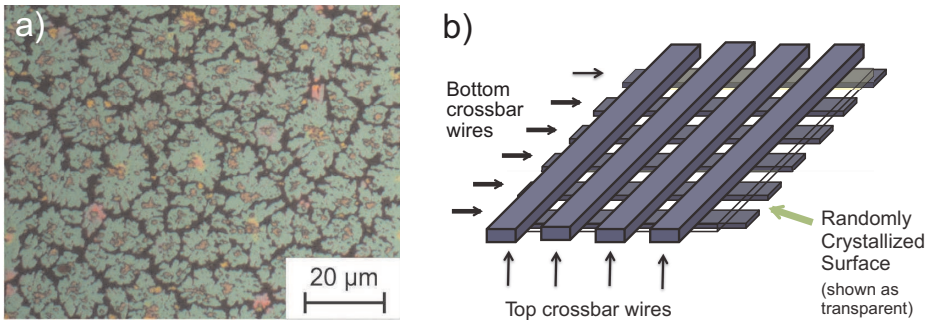

Fig. 1. a) A polycrystalline film resulting from the ALILE process, illustrating the high entropy and disorder in the structure. The green areas are silicon crystallites, possessing a random distribution and strongly irregular shape. b) The schematics of the crossbar read-out circuitry.

Inherently Slow Read-Out Speed. Up to now, we have mainly described a memory-like structure with a high information content and density. Also large arrays of SRAM cells or Butterfly PUFs could fulfill these criteria, albeit presumably at lower information densities. The perhaps most unusual characteristic of Crossbar PUFs is that they promise to guarantee an inherently slow read-out speed [28]. To achieve this property, the Crossbar PUF must be built in one large, monolithic block, not from separate blocks as modern semiconductor memories. The wires are intentionally designed to have only a low, limited current-carrying capacity. Simulations conducted in [28] showed that in such large blocks, depending on the fabrication parameters, several milliseconds must elapse before the sense current/voltage stabilizes. This leads to read-out speeds of around 100 bits/sec [28].

The two apparent strategies to accelerate read-out would be to increase the sense current/voltage, or to conduct a parallel read-out at several crossings. But both approaches lead to a higher current load in the monolithic crossbar, which is proportional

to the achieved speed up. They therefore quickly overload and destroy the limited wires [28]. Removing the original wires of the crossbar, which very densely cover the whole crystallized system, and replacing them with a faster read-out mechanism seems practically infeasible without destroying the PUF's structure and current-voltage characteristics. This makes the PUF's original responses unreadable [28].

3 The Protocol

We now provide a protocol for $\binom{2}{1}$-OT on the basis of Strong PUFs, which is inspired by techniques originally presented in [14]. Since SHIC PUFs are a subclass of Strong PUFs, the protocol works for both PUF types interchangeably — using SHIC PUFs only causes some small advantages in the resulting security features (see section 3.3). As a subprotocol, we employ interactive hashing [13] [14].

3.1 Interactive Hashing

In a nutshell, interactive hashing [13] is a cryptographic two-player protocol, in which Alice has no input, and Bob's initial input is an m-bit string S. At the end of the protocol, Alice knows two m-bit strings U_1 and U_2, with the properties that (i) $U_b = S$ for some bit $b \in \{0, 1\}$, but Alice does not know the value of b, and that (ii) the other string $U_{b\oplus 1}$ is an essentially random bitstring of length m, which neither Alice nor Bob can determine alone. A protocol for interactive hashing can be constructed as follows.

Protocol 1: INTERACTIVE HASHING

Prerequisites:

1. Alice holds no input, Bob holds an m-bit string S as input.
2. Let G be the following class of 2-universal hash functions:

$$G = \{g(x) = a * x \mid a \text{ is an element of the set } \{0,1\}^m\},$$

where $*$ denotes the scalar product between the vectors a and x.

Protocol:

The protocol consists of $m-1$ rounds. In the j-th round, for $j = 1, \ldots, m-1$, Alice executes the following steps:

1. Alice chooses a function g_j uniformly at random from the set G. Let the m-ary binary vector a_j be the description of G. If a_j is linearly dependent on the $a_1, \ldots, a_{m-1}$, then Alice repeats step 1 until a_j is linearly independent.
2. Alice announces g_j to Bob.
3. Bob computes $b_j = g_j(S) = a_j * S$ and sends b_j to Alice.

At the end of the protocol, Alice knows $m-1$ linear equations satisfied by S. Since the a_j's are linearly independent, there are exactly two different m-bit strings U_1 and U_2 that satisfy the system of equations set up by Bob. These solutions can be found by Alice via standard linear algebra. U_1 and U_2 have the property that exactly one of them is equal to S, but obviously Alice has no chance in telling which one it is. For further details see [13] [14].

3.2 Oblivious Transfer

Protocol 2: $\binom{2}{1}$-OBLIVIOUS TRANSFER BY STRONG PUFS

Prerequisites:

1. Bob holds a Strong PUF S. We assume without loss of generality that the responses R_C^S of S consist of a single bit. [3]
2. Alice and Bob have agreed on an encoding scheme $E(\cdot)$ with the following properties:
   (a) $E(\cdot)$ efficiently encodes finite tuples of PUF-challenges C_i of the form $T = (C_1, \ldots, C_k)$ as finite binary strings.
   (b) $E(\cdot)$ is reversed by a decoding scheme $D(\cdot)$, such that $E(D(T)) = T$ for all tuples T of the form $T = (C_1, \ldots, C_k)$ (with the C_i being challenges of S).
   (c) $D(\cdot)$ uniquely associates a tuple $T = D(x)$ with any finite binary string x.

   Similar encoding schemes can be found, for example, in [32] or [14].
3. Alice holds two bits b_0 and b_1, which are unknown to Bob.
4. Bob holds a choice bit c, which is unknown to Alice.

Protocol:

1. Bob chooses a tuple of challenges $T = (C_1, \ldots, C_n)$ uniformly at random, and determines the corresponding responses $R_{C_1}, \ldots, R_{C_n}$.
2. Bob sends or transfers the Strong PUF S to Alice.
3. Alice and Bob get engaged in an interactive hashing protocol, where Bob's input is $E(T)$.
4. The output of this interactive hashing protocol, which is both known to Alice and Bob, are two strings U_0 and U_1. One of these strings U_0, U_1 is equal to $E(T)$. Let us call the index of that string i_0, i.e. $U_{i_0} = E(T)$.

   Note: Bob knows i_0, since he knows both U_0, U_1 and $E(T)$.

5. Bob sets the bit $c' = i_0 \oplus c$, and sends c' to Alice.
6. Alice determines by measurement on the PUF S the values

$$R_{Z_1}, \ldots, R_{Z_n},$$

   where the Z_i are the elements of the tuple $D(U_{c'})$ (which, by the properties of $D(\cdot)$, are all challenges of S). Furthermore, she determines by measurement on S the values

$$R_{Z_1'}, \ldots, R_{Z_n'},$$

   where the $Z_i{}'$ are the elements of the set $D(U_{c' \oplus 1})$.

[3] If a response consists of multiple bits $b_1 \cdots b_k$, we can, for example, take the XOR of all these bits, or employ fuzzy extractors.

Note: At this point of the protocol, Alice has chosen two sets of PUF-responses $R_{Z_1}, \ldots, R_{Z_n}$ and $R_{Z_1}{}', \ldots, R_{Z_n}{}'$. Bob knows exactly one of these sets, namely the one that is equal to $R_{C_1}, \ldots, R_{C_n}$. The other set is unknown to Bob. Furthermore, Alice does not know which of the two sets of responses is known to Bob.

7. Alice forms the two strings s_0 and s_1 according to the following rules:

$$s_0 = b_0 + R_{Z_1} + \ldots + R_{Z_n} \mod 2,$$

and

$$s_1 = b_1 + R_{Z_1}{}' + \ldots + R_{Z_n}{}' \mod 2.$$

8. Alice sends s_0 and s_1 to Bob.
9. Bob obtains the bit b_c he selected through his choice bit c as

$$b_c = s_c + R_{C_1} + \ldots + R_{C_n} \mod 2.$$

3.3 Discussion

Security. The security of the protocol depends on the fact that Bob does not know both sets $R_{Z_1}, \ldots, R_{Z_n}$ and $R_{Z_1}{}', \ldots, R_{Z_n}{}'$ in step 7. If he did, then he could learn both bits b_0 and b_1. This is where property (iii) (see page 3) of Strong PUFs and SHIC PUFs becomes relevant. Due to this property, Bob cannot know all CRPs of the Strong PUF/SHIC PUF, but only a fraction γ with $0 < \gamma < 1$. Since one of the sets $R_{Z_1}, \ldots, R_{Z_n}$ and $R_{Z_1}{}', \ldots, R_{Z_n}{}'$ is chosen at random in the interactive hashing protocol, the probability that Bob knows the corresponding CRPs is γ^n, i.e. it is exponentially low in the security parameter n of the protocol. The fact that Alice does not learn Bob's choice bit c stems from the security properties of the interactive hashing protocol, which prevents that Alice learns which of the two strings U_1 or U_2 is equal to Bob's private input S [13] [14].

The security difference in using Strong PUFs and SHIC PUFs in Protocol 2 is that by its definition and property (vi), a secure SHIC PUF would fulfill the essential requirement (iii) (see page 3) independent of the computational power of the adversary. Secure SHIC PUFs hence could guarantee the protocol's security also against cheating parties with unlimited computational potential.

Practicality. The communication and storage requirements are mild: Bob must store only n CRPs, and the protocol has around m rounds for the interactive hashing. The latter can be reduced to a constant the techniques described in [15].

In order to cope with potential noise in the PUF responses, presumably standard PUF error correction such as helper data (see [2] [27] and references therein) could be used. In that case, a few steps of the protocol should be adjusted. Firstly, Bob measures and stores noisy data R_{C_i} in Step 1. Alice likewise obtains noisy responses R_{Z_i} and $R_{Z_i}{}'$ in Step 6 of the protocol, and extracts helper data W_{Z_i} and $W_{Z_i}{}'$, together with secrets S_{Z_i} and $S_{Z_i}{}'$. In Step 7, Alice uses the secrets S_{Z_i} and $S_{Z_i}{}'$ (instead of the values R_{Z_i} and $R_{Z_i}{}'$) to "encrypt" the bits b_0 and b_1. In Step 8, she transmits the

corresponding helper data W_{Z_i} and $W_{Z_i{}'}$ together with the strings s_0 and s_1. Of these two sets of helper data, Bob uses the one that matches his data set R_{C_i}. He derives identical secrets as Alice from the R_{C_i}, and uncovers the bit b_c from $s_{i_0 \oplus c}$.

Symmetry. Oblivious transfer is known to be a symmetric primitive [31]: Given an OT protocol where Alice is the sender and Bob is the receiver, one can construct the "reverse" OT protocol where Alice acts as receiver and Bob as sender. The construction of [31] is generic, and independent of the concrete implementation of the OT.

Therefore, Protocol 2 can also be used to implement OT in the other direction, i.e. from Bob to Alice, without re-transferring the PUF from Alice to Bob. This is an important practicality asset: In many applications, the physical transfer of the PUF in one direction is executed naturally (e.g. in a hardware shipped from a manufacturer to a customer, or on a bank card carried to an automated teller machine (ATM) by a customer). Once accomplished, this allows oblivious transfer in both directions and secure two-party computations, e.g. between the manufacturer and the hardware.

4 Summary

We discussed a protocol for oblivious transfer on the basis of Strong PUFs and SHIC PUFs. It allows OT and secure two-party computation between two players, provided that (i) Player A had previous access to the PUF, and (ii) only Player B holds physical possession of the PUF at the time of the protocol execution. These circumstances occur frequently in practice, for example between a central authority on the one hand and mobile hardware systems, decentral terminals, or security tokens (including bank cards, ID cards, access cards, and the like) on the other hand. The protocol does not use any computational assumptions other than the security of the PUF.

References

1. R. Pappu, B. Recht, J. Taylor, N. Gershenfeld: *Physical One-Way Functions*, Science, vol. 297, pp. 2026-2030, 20 September 2002.
2. P. Tuyls, B. Skoric: *Strong Authentication with Physical Unclonable Functions.* In: Security, Privacy and Trust in Modern Data Management, M. Petkovic, W. Jonker (Eds.), Springer, 2007.
3. M. O. Rabin: *How to exchange secrets by oblivious transfer.* Technical Report TR-81, Harvard University, 1981.
4. S. Even, O. Goldreich, A. Lempel: *A randomized protocol for signing contracts.* In: Proc. CRYPTO 82 (R. L. Rivest, A. Sherman, S. Chaum, Eds.), pp. 205-210, Plenum Press, 1983.
5. C. Crepeau: *Equivalence between two flavors of oblivious transfer.* CRYPTO '87 (C. Pomerance, Ed.), LNCS Vol. 293, pp. 350–354, Springer, 1988.
6. A. C.-C. Yao: *How to generate and exchange secrets.* Proc. of the 27th IEEE Symposium on the Foundations of Computer Science (FOCS), pp. 162–167, 1986.
7. O. Goldreich, S. Micali, A. Widgerson: *How to play any mental game, or a completeness theorem for protocols with honest majority.* Proc. of the 19th Annual Symposium on the Theory of Computing (STOC), pp. 218–229, 1987.

8. O. Goldreich, R. Vainish: *How to solve any protocol problem – an efficiency improvement.* CRYPTO 87 (C. Promenance, Ed.), LNCS Vol. 293, pp. 73-86, Springer 1988.
9. J. Kilian: *Founding cryptography on oblivious transfer*. Proceedings, 20th Annual ACM Symposium on the Theory of Computation (STOC), 1988.
10. C. Crepeau, J. van de Graaf, A. Tapp: *Committed oblivious transfer and private multi-party computations*. CRYPTO 95, LNCS Vol. 963, pp. 110-123, Springer 1995.
11. G.P. He, Z.D. Wang: *Oblivious transfer using quantum entanglement*. Physical Review A 2006, VOL 73; NUMB 1; PART A, pages 012331.
12. S. Wehner, C. Schaffner, B.M. Terhal, *Cryptography from noisy storage*. Phys Rev Lett. 2008 Jun 6;100(22):220502.
13. M. Naor, R. Ostrovsky, R. Venkatesan, M. Yung: *Perfect zero-knowledge arguments for NP using any one-way function*. Journal of Cryptology 11 (1998), no. 2, 87–108.
14. C. Cachin, C. Crepeau, J. Marcil: *Oblivious transfer with a memory-bounded receiver.* Proceeding of the 39th Annual Symposium on Foundations of Computer Science, 1998.
15. Y.Z. Ding, D. Harnik, A. Rosen, R. Shaltiel: *Constant-Round Oblivious Transfer in the Bounded Storage Model*. Journal of Cryptology, 2007.
16. C. Crepeau: *Efficient cryptographic protocols based on noisy channels*. EUROCRYPT 97 (Walter Fumy, Ed.), LNCS Vol. 1233, pp. 306–317, Springer, 1997.
17. Jürg Wullschleger: *Oblivious Transfer from Weak Noisy Channels.* TCC 2009: 332-349
18. Jorge Guajardo, Sandeep S. Kumar, Geert Jan Schrijen, Pim Tuyls: *FPGA Intrinsic PUFs and Their Use for IP Protection*. CHES 2007: 63-80
19. Blaise Gassend, *Physical Random Functions*, MSc Thesis, MIT, 2003.
20. Daihyun Lim: *Extracting Secret Keys from Integrated Circuits.* MSc Thesis, MIT, 2004.
21. B. Gassend, D. Lim, D. Clarke, M. v. Dijk, S. Devadas: *Identification and authentication of integrated circuits.* Concurrency and Computation: Practice & Experience, pp. 1077 - 1098, Volume 16, Issue 11, September 2004.
22. J.-W. Lee, D. Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas. *A technique to build a secret key in integrated circuits with identification and authentication applications.* In Proceedings of the IEEE VLSI Circuits Symposium, June 2004.
23. M. Majzoobi, F. Koushanfar, M. Potkonjak: *Lightweight Secure PUFs.* IC-CAD 2008.
24. G. Csaba, X. Ju, Z. Ma, Q. Chen, W. Porod, J. Schmidhuber, U. Schlichtmann, P. Lugli, U. Rührmair: *Application of Mismatched Cellular Nonlinear Networks for Physical Cryptography*. IEEE CNNA, 2010.
25. M. Majzoobi, F. Koushanfar, M. Potkonjak: *Testing Techniques for Hardware Security.* IEEE International Test Conference, 2008.
26. U. Rührmair, F. Sehnke, J. Soelter, S. Devadas, J. Schmidhuber: *Modeling Attacks on Physical Unclonable Functions.* Submitted, 2010.
27. U. Rührmair, C. Jaeger, C. Hilgers, M. Algasinger, G. Csaba, M. Stutzmann: *Security Applications of Diodes with Random Current-Voltage Characteristics.* Financial Cryptography and Data Security, 2010.
28. U. Rührmair, C. Jaeger, M. Bator, M. Stutzmann, P. Lugli, and G. Csaba: *Applications of High-Capacity Crossbar Memories in Cryptography.* To appear in IEEE Transactions on Nanotechnology, 2010.
29. C. Jaeger, M. Algasinger, U. Rührmair, G. Csaba, M. Stutzmann: *Random pn-junctions for physical cryptography.* To appear in Applied Physics Letters, 2010.
30. G. E. Suh, S. Devadas: *Physical Unclonable Functions for Device Authentication and Secret Key Generation.* DAC 2007: 9-14
31. S. Wolf, J. Wullschleger: *Oblivious Transfer Is Symmetric.* EUROCRYPT 2006: 222-232
32. T. M. Cover: *Enumerative Source Encoding*. IEEE Transactions on Information Theory 19 (1973), No. 1, pp. 73 – 77.