J Cryptogr Eng (2013) 3:17–28
DOI 10.1007/s13389-013-0052-8

CHES 2012

On the practical use of physical unclonable functions in oblivious transfer and bit commitment protocols

Ulrich Rührmair · Marten van Dijk

Received: 18 November 2012 / Accepted: 15 January 2013 / Published online: 26 March 2013

Abstract In recent years, PUF-based schemes have been suggested not only for the basic tasks of tamper-sensitive key storage or the identification of hardware systems, but also for more complex protocols like oblivious transfer (OT) or bit commitment (BC), both of which possess broad and diverse applications. In this paper, we continue this line of research. We first present an attack on two recent OT and BC protocols which have been introduced by Brzuska et al. (CRYPTO, LNCS 6841, pp 51–70, Springer 2011). The attack quadratically reduces the number of CRPs which malicious players must read out to cheat, and fully operates within the original communication model of Brzuska et al. (CRYPTO, LNCS 6841, pp 51–70, Springer 2011). In practice, this leads to insecure protocols when electrical PUFs with a medium challenge-length are used (e.g., 64 bits), or whenever optical PUFs are employed. These two PUF types are currently among the most popular designs of so-called Strong PUFs. Secondly, we show that the same attack applies to a recent OT protocol of Ostrovsky et al. (IACR Cryptol. ePrint Arch. 2012:143, 2012), leading to exactly the same consequences. Finally, we discuss countermeasures. We present a new OT protocol with better security properties, which utilizes interactive hashing as a substep and is based on an earlier protocol by Rührmair (TRUST, LNCS 6101, pp 430–440, Springer 2010). We then closely analyze its properties, including its security, security amplification, and practicality.

U. Rührmair (✉)
Computer Science Department, Technische Universität München, 80333 Munich, Germany
e-mail: ruehrmair@ilo.de

M. van Dijk (✉)
MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA, USA
e-mail: martenvdijk@gmail.com

Keywords Physical unclonable functions (PUFs) · Cryptographic protocols · Oblivious transfer · Bit commitment · Security analysis · Interactive hashing

1 Introduction

Today's electronic devices are mobile, cross-linked and pervasive, which makes them an accessible target for adversaries. The well-known protective cryptographic techniques all rest on the concept of a secret binary key: they presuppose that devices store a piece of digital information that is, and remains, unknown to an adversary. It turns out that this requirement is difficult to realize in practice. Physical attacks such as invasive, semi-invasive or side-channel attacks carried out by adversaries with one-time physical access to the devices, as well as software attacks like application programming interface (API) attacks, viruses or Trojan horses, can lead to key exposure and security breaks. As Ron Rivest emphasized in his keynote talk at CRYPTO 2011 [25], merely calling a bit string a "secret key" does not make it secret, but rather identifies it as an interesting target for the adversary.

Indeed, one main motivation for the development of *physical unclonable functions (PUFs)* was their promise to better protect secret keys. A PUF is an (at least partly) disordered physical system P that can be challenged with so-called external stimuli or challenges c, upon which it reacts with corresponding responses r. Contrary to standard digital systems, these responses depend on the micro- or nanoscale structural disorder of the PUF. 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. Any PUF P thus implements a unique and individual function f_P that maps challenges c to responses $r = f_P(c)$.

The tuples (c, r) are called the challenge–response pairs (CRPs) of the PUF.

Due to its complex internal structure, a PUF can avoid some of the shortcomings of classical digital keys. It is usually harder to read out, predict, or derive PUF responses than to obtain digital keys that are stored in non-volatile memory. The PUF responses are only generated when needed, which means that no secret keys are present permanently in the system in an easily accessible digital form. Finally, certain types of PUFs are naturally tamper sensitive: their exact behavior depends on minuscule manufacturing irregularities, often in different layers of the IC, and removing or penetrating these layers will automatically change the PUF's read-out values. These facts have been exploited in the past PUF research for different PUF-based security protocols. Prominent examples include identification [10,24], key exchange [24], and various forms of (tamper sensitive) key storage and applications thereof, such as intellectual property protection or read-proof memory [12,15,35].

In recent years, the use of PUFs in more advanced cryptographic protocols together with formal security proofs has been investigated. In these protocols, usually a PUF-type known as Strong PUFs is employed [12,27,31,32].[1] These PUFs have a very large challenge set and a freely accessible challenge–response interface. They are used similar to a "physical random oracle", which is transferred between the parties, and which can be read out exactly by the very party who currently holds physical possession of it. Their input–output behavior is assumed to be so complex that its response to a randomly chosen challenge cannot be predicted numerically and without direct physical measurement, not even by a person who had physical access to the PUF at earlier points in time. In 2010, Rührmair [26] showed that oblivious transfer (OT) can be realized between two parties by physically transferring a PUF in this setting. He observed that via the classical reductions of Kilian [14], this implies PUF-based bit commitment and PUF-based secure multi-party computations. In the same year, the first formal security proof for a PUF protocol was provided by Rührmair, Busch and Katzenbeisser [27]. They presented definitions and a reductionist security proof for PUF-based identification protocols. At CRYPTO 2011 Brzuska et al. [2] adapted Canetti's universal composition (UC) framework [4] to include PUFs. They gave PUF-based protocols for oblivious transfer (OT), bit commitment (BC) and key exchange (KE) and proved them to be secure in their framework. Just recently, Ostrovsky et al. [20] revisited the use of PUFs in the UC framework, discussing new protocols and attack models. An overview of the area of PUFs and related structures has been given by Rührmair et al. [28].

The investigation of advanced cryptographic settings for PUF makes sense even from the perspective of a pure practitioner: first, it clarifies the potential of PUFs in theory, a necessary prerequisite before this potential can be unleashed in commercial applications without risking security failures. Secondly, BC and OT protocols are extremely versatile cryptographic primitives, which allow the implementation of such diverse tasks as zero-knowledge identification, the enforcement of semi-honest behavior in cryptographic protocols, secure multi-party computation (including online auctions or electronic voting), or key exchange. If these tasks shall be realized securely in practice by PUFs, a theoretical investigation of the underlying primitives—in this case BC and OT—is required first.

In this paper, we continue this line of research, and revisit the use of PUFs in OT and BC protocols. Particular emphasis is placed on the achievable *practical security* if some well-known PUFs (like electrical PUFs with 64-bit challenge lengths or optical PUFs) are used in the protocols. We start by observing an attack on the OT and BC protocols of Brzuska et al. [2,3] which quadratically reduces the number of responses that a malicious player must read out to cheat. It works fully in the original communication model of Brzuska et al. and makes no additional assumptions. As we show, the attack makes the protocols insecure in practice if electrical PUFs with medium bitlengths around 64 bits are used, and generally if optical PUFs are employed. This has a special relevance since the use of optical PUFs for their protocols had been explicitly proposed by Brzuska et al. (see Sect. 8 of [3]). Secondly, we investigate countermeasures against our attack, and show that interactive hashing can be used to enhance the security of PUF-based OT and BC protocols.

Our work continues the recent trend of a formalization of PUFs, including the formalization of PUF security features [1,2,7,20,27,32], cryptanalyses of PUF protocols [29,30], more detailed investigations of non-trivial communication settings [7,20], and formal security proofs [2,7,20,27]. This trend will eventually lay the foundations for future PUF research, and seems indispensible for a healthy long-term development of the field. It also combines protocol design and practical security analyses in a novel manner.

Organization of this paper. In Sect. 2 we present the protocols of Brzuska et al. [2] to achieve a self-contained treatment. Section 3 gives our quadratic attack. Section 4 discusses its practical effect. Section 5 shows the applicability of our attack to a recent OT protocol by Ostrovsky et al. [20]. Section 6 discusses one particular countermeasure. Section 7 gives a new, PUF-based protocol for OT based on interactive

[1] Strong PUFs have sometimes also been referred to as *physical random function* [10] in the literature. We emphasize that the Weak/Strong PUF terminology, which was originally introduced by Guajardo et al. [12], is not to be understood in a judgmental or pejorative manner.

hashing, and analyzes its security. We conclude the paper in Sect. 8.

2 The protocols of Brzuska et al.

Our aim in this paper is to present a quadratic attack on two recent PUF protocols for OT and BC by Brzuska et al. [2,3] and to discuss its practical relevance. To achieve a self-contained treatment, we will now present these two protocols. To keep our exposition simple, we will not use the full UC notation of [2], and will present the schemes mostly without error correction mechanisms, since the latter play no role in the context of our attack.

The protocols use two communication channels between the communication partners: a binary channel, over which all digital communication is handled. It is assumed that this channel is non-confidential, but authenticated. And secondly an insecure physical channel, over which the PUF is sent. It is assumed that adversaries can measure adaptively selected CRPs of the PUF while it is in transition over this channel.

2.1 Oblivious transfer

The OT protocol of [2] implements one-out-of-two string oblivious transfer. It is assumed that in each subsession the sender P_i initially holds two (fresh) bitstrings $s_0, s_1 \in \{0,1\}^\lambda$, and that the receiver P_j holds a (fresh) choice bit b.

Brzuska et al. generally assume in their treatment that after error correction and the application of fuzzy extractors, a PUF can be modeled as a function $\mathsf{PUF} : \{0,1\}^\lambda \to \{0,1\}^{rg(\lambda)}$. We use this model throughout this paper, too. In the subsequent protocol of Brzuska et al., it is furthermore assumed that $rg(\lambda) = \lambda$, i.e., that the PUF implements a function $\mathsf{PUF} : \{0,1\}^\lambda \to \{0,1\}^\lambda$ (compare [2,3]).

Protocol 1 PUF-based oblivious transfer ([2], slightly simplified description)

External parameters: The protocol has a number of external parameters, including the security parameter λ, the session identifier `sid`, a number N that specifies how many subsessions are allowed, and a pre-specified PUF-family $\mathcal{P}$, from which all PUFs which are used in the protocol must be drawn.

Initialization phase: Execute once with fixed session identifier `sid`:

1. The receiver holds a PUF which has been drawn from the family $\mathcal{P}$.
2. The receiver measures l randomly chosen CRPs $c_1, r_1, \ldots, c_l, r_l$ from the PUF, and puts them in a list $\mathcal{L} := (c_1, r_1, \ldots, c_l, r_l)$.
3. The receiver sends the PUF to the sender.

Subsession phase: Repeat at most N times with fresh subsession identifier `ssid`:

1. The sender's inputs are two strings $s_0, s_1 \in \{0,1\}^\lambda$, and the receiver's input is a bit $b \in \{0,1\}$.
2. The receiver chooses a CRP (c, r) from the list $\mathcal{L}$ at random.
3. The sender chooses two random bitstrings $x_0, x_1 \in \{0,1\}^\lambda$ and sends x_0, x_1 to the receiver.
4. The receiver returns the value $v := c \oplus x_b$ to the sender.
5. The sender measures the responses r_0 and r_1 of the PUF that correspond to the challenges $c_0 := v \oplus x_0$ and $c_1 := v \oplus x_1$.
6. The sender sets the values $S_0 := s_0 \oplus r_0$ and $S_1 := s_1 \oplus r_1$, and sends S_0, S_1 to the receiver.
7. The receiver recovers the string s_b that depends on his choice bit b as $s_b = S_b \oplus r$. He erases the pair (c, r) from the list $\mathcal{L}$.

Comments. The protocol implicitly assumes that the sender and receiver can interrogate the PUF whenever they have access to it, i.e., that the PUF's challenge–response interface is publicly accessible and not protected. This implies that the employed PUF must possess a large number of CRPs. Using a PUF with just a few challenges does not make sense: the receiver could then create a full look-up table for all CRPs of such a PUF before sending it away in Step 3 of the initialization phase. This would subsequently allow him to recover both strings s_0 and s_1 in Step 6 of the protocol subsession, as he could obtain r_0 and r_1 from his look-up table. Similar observations hold for the upcoming protocol 2. Indeed, all protocols discussed in this paper require PUFs with a large number of challenges and publicly accessible challenge–response interfaces. These PUFs are usually referred to as *physical random functions* [9] or simply as *Strong PUFs* in the literature [12,27,31,32].

Furthermore, please note that no physical transfer of the PUF is envisaged during the subsessions of the protocol. According to the model of Brzuska et al., an adversary only has access to it during the initialization phase, but not between the subsessions. This protocol use has some similarities with a stand-alone usage of the PUF, in which exactly one PUF-transfer occurs between the parties.

2.2 Bit commitment

The second protocol of [2] implements PUF-based bit commitment (BC) by a generic reduction to PUF-based OT. The BC sender initially holds a bit b. When the OT protocol is called as a subprotocol, the roles of the sender and receiver are reversed: the BC sender acts as the OT receiver, and the BC receiver as the OT sender. The details are as follows.

Protocol 2 PUF-based BC via PUF-based OT ([2], slightly simplified description)

Commit phase:

1. The BC sender and the BC receiver jointly run an OT protocol (for example Protocol 1).
   (a) In this OT protocol, the BC sender acts as OT receiver and uses his bit b as the choice bit of the OT protocol.
   (b) The BC receiver acts as OT sender. He chooses two strings $s_0, s_1 \in \{0,1\}^\lambda$ at random, and uses them as his input s_0, s_1 to the OT protocol.
2. When the OT protocol is completed, the BC sender has learned the string $v := s_b$. This closes the commit phase.

Reveal phase:

1. To reveal bit b, the BC sender sends the string (b, v) (with $v = s_b$) to the BC receiver.

Comments. The security of the BC protocol is inherited from the underlying OT protocol. Once this protocol is broken, the security of the BC protocol is also lost. This will be relevant in the upcoming sections.

3 A quadratic attack on Protocols 1 and 2

We will now discuss a cheating strategy in Protocols 1 and 2. Compared to an attacker who exhaustively queries the PUF for all of its m possible challenges, we describe an attack on Protocols 1 and 2 which reduces this number to $\sqrt{m}$. As we will argue later in Sect. 4, this has a particularly strong effect on the protocol's security if an optical PUF is used (as has been explicitly suggested by [3]), or if electrical PUFs with medium challenge lengths of 64 bits are used.

Our attack rests on the following lemma.

Lemma 3 *Consider the vector space* $(\{0,1\}^\lambda, \oplus)$, $\lambda \geq 2$, *with basis* $\mathcal{B} = \{a_1, \ldots, a_{\lfloor\lambda/2\rfloor}, b_1, \ldots, b_{\lceil\lambda/2\rceil}\}$. *Let* A *be equal to the linear subspace generated by the vectors in* $\mathcal{B}_A = \{a_1, \ldots, a_{\lfloor\lambda/2\rfloor}\}$, *and let* B *be the linear subspace generated by the vectors in* $\mathcal{B}_B = \{b_1, \ldots, b_{\lceil\lambda/2\rceil}\}$. *Define* $M := A \cup B$. *Then it holds that:*

(i) Any vector $z \in \{0,1\}^\lambda$ *can be expressed as* $z = a \oplus b$ *with* $a, b \in M$, *and this expression (i.e., the vectors* a *and* b*) can be found efficiently (i.e., in at most poly*(λ) *steps).*

(ii) For all distinct vectors $x_0, x_1, v \in \{0,1\}^\lambda$ *there is an equal number of combinations of linear subspaces* A *and* B *as defined above for which* $x_0 \oplus v \in A$ *and* $x_1 \oplus v \in B$.

(iii) M *has cardinality* $|M| \leq 2 \times 2^{\lceil\lambda/2\rceil}$.

Proof (i) Notice that any vector $z \in \{0,1\}^\lambda$ can be expressed as a linear combination of all basis vectors: $z = \sum u_i a_i + \sum v_j b_j$, i.e., $z = a \oplus b$ with $a \in A$ and $b \in B$. This expression is found efficiently using Gaussian elimination.

(ii) Without loss of generality, since x_0, x_1 and v are distinct vectors, we may choose $a_1 = x_0 \oplus v \neq 0$ and $b_1 = x_1 \oplus v \neq 0$. The number of combinations of linear subspaces A and B is independent of the choice of a_1 and b_1 (notice that if $x_0 \neq x_1$ but $v = x_0$, then the number of combinations is twice as large).

(iii) The bound follows from the construction of M and the cardinalities of A and B, which are $|A| = 2^{\lfloor\lambda/2\rfloor}$ and $|B| = 2^{\lceil\lambda/2\rceil}$. □

An example. Let us give an example to illustrate the principle of Lemma 3. Consider the vector space $(\{0,1\}^\lambda, \oplus)$ for an even λ, and choose as subbases $\mathcal{B}_{A_0} = \{e_1, \ldots, e_{\lambda/2}\}$ and $\mathcal{B}_{B_0} = \{e_{\lambda/2+1}, \ldots, e_\lambda\}$, where e_i is the unit vector of length λ that has a one in position i and zeros in all other positions. Then the basis $\mathcal{B}_{A_0}$ spans the subspace A_0 that contains all vectors of length λ whose second half is all zero, It then follows immediately that every vector $z \in \{0,1\}^\lambda$ can be expressed as $z = a \oplus b$ with $a \in A_0$ and $b \in B_0$, or, saying this differently, with $a, b \in M$ and $M := A_0 \cup B_0$. It is also immediate that M has cardinality $|M| \leq 2 \times 2^{\lambda/2}$.

Relevance for PUFs. The lemma translates into a PUF context as follows. Suppose that a malicious and an honest player play the following game. The malicious player gets access to a PUF with challenge length λ in an initialization period, in which he can query CRPs of his choice from the PUF. After that, the PUF is taken away from him. Then, the honest player chooses a vector $z \in \{0,1\}^\lambda$ and sends it to the malicious player. The malicious player wins the game if he can present the correct PUF responses r_0 and r_1 to two arbitrary challenges c_0 and c_1 which have the property that $c_0 \oplus c_1 = z$. Our lemma shows that to win the game with certainty, the malicious player does not need to read out the entire CRP space of the PUF in the initialization phase; he merely needs to know the responses to all challenges in the set M of Lemma 3, which has a quadratically reduced size compared to the entire CRP space. This observation is at the heart of the attack described below.

To make the attack hard to detect for the honest player, it is necessary that the attacker chooses random subspaces A and B, and does not use the above trivial choices A_0 and B_0 all the time. This fact motivates the random choice of A and B in Lemma 3. The further details are as follows.

The attack. As in [2,3], we assume that the PUF has got a challenge set of $\{0,1\}^\lambda$. Given Lemma 3, the OT receiver (who initially holds the PUF) can achieve a quadratic advantage in Protocol 1 as described below.

First, he chooses uniformly random linear subspaces A and B, and constructs the set M, as described in Lemma 3. While he holds possession of the PUF before the start of the protocol, he reads out the responses to all challenges in M. Since $|M| \leq 2 \times 2^{\lceil \lambda/2 \rceil}$, this is a quadratic improvement over reading out all responses of the PUF.

Next, he starts the protocol as normal. When he receives the two values x_0 and x_1 in Step 3 of the protocol, he computes two challenges c_0^* and c_1^* both in set M such that

$$x_0 \oplus x_1 = c_0^* \oplus c_1^*.$$

According to Lemma 3(i), this can be done efficiently (i.e., in $poly(\lambda)$ operations). Notice that, since the receiver knows all the responses corresponding to challenges in M, he in particular knows the two responses r_0^* and r_1^* that correspond to the challenges c_0^* and c_1^*.

Next, the receiver deviates from the protocol and sends the value $v := c_0^* \oplus x_0$ in Step 4. For this choice of v, the two challenges c_0 and c_1 that the sender uses in Step 5 satisfy

$$c_0 := c_0^* \oplus x_0 \oplus x_0 = c_0^*$$

and

$$c_1 := c_0^* \oplus x_0 \oplus x_1 = c_0^* \oplus c_0^* \oplus c_1^* = c_1^*.$$

By Lemma 3(ii), Alice cannot distinguish the received value v in Step 4 from any random vector v. In other words, Alice cannot distinguish Bob's malicious behavior (i.e., fabricating a special v with suitable properties) from honest behavior. As a consequence, Alice continues with Step 6 and transmits $S_0 = s_0 \oplus r_0^*$ and $S_1 = s_1 \oplus r_1^*$. Since Bob knows both r_0^* and r_1^*, he can recover both s_0 and s_1. This breaks the security of the protocol.

Please note the presented attack is simple and effective: it fully works within the original communication model of Brzuska et al. [2,3]. Furthermore, it does not require laborious computations of many days on the side of the attacker (as certain modeling attacks on PUFs do [31]). Finally, due to the special construction we proposed, the honest players will not notice the special choice of the value v, as the latter shows no difference from a randomly chosen value.

Effect on bit commitment (Protocol 2). Due to the reductionist construction of Protocol 2, our attack on the oblivious transfer scheme of Protocol 1 directly carries over to the bit commitment scheme of Protocol 2 if Protocol 1 is used in it as a subprotocol. Using the attack, a malicious sender can open the commitment in both ways by reading out only $2 \times 2^{\lceil \lambda/2 \rceil}$ responses (instead of all 2^λ responses) of the PUF. On the other hand, it can be observed easily that the hiding property of the BC Protocol 2 is unconditional, and is not affected by our attack.

4 Practical consequences of the attack

What are the practical consequences of our quadratic attack, and how relevant is it in real-world applications? The situation can perhaps be illustrated via a comparison to classical cryptography. What effect would a quadratic attack have on schemes like RSA, DES and SHA-1? To start with RSA, the effect of a quadratic attack here is rather mild: the length of the modulus must be doubled. This will lead to longer computation times, but restore security without further ado. In the case of single-round DES, however, a quadratic attack would destroy its security, and the same holds for SHA-1. The actual effect of our attack on PUF-based OT and BC has some similarities with DES or SHA-1: PUFs are finite objects, which cannot be scaled in size indefinitely due to area requirements, arising costs, and stability problems. This will also become apparent in our subsequent discussion.

4.1 Electrical integrated PUFs

We start our discussion by electrical integrated PUFs, and take the well-known Arbiter PUF as an example. It has been discussed in theory and realized in silicon mainly for challenge lengths of 64 bits up to this date [10,11,16,34]. Our attack on such a 64-bit implementation requires the read-out of $2 \times 2^{32} = 8.58 \times 10^9$ CRPs by the receiver. This read-out can be executed before the protocol (i.e., not during the protocol), and will hence not be noticed by the sender. Assuming a MHz CRP read-out rate [16] of the Arbiter PUF, the read-out takes 8.58×10^3 s, or less than 144 min.

Please note that the attack is independent of the cryptographic hardness of the PUF, such as its resilience against machine learning attacks. For example, a 64-bit, 8-XOR-Arbiter PUF (i.e., an Arbiter PUF with eight parallel standard 64-bit Arbiter PUFs whose single responses are XORed at the end of the structure) is considered secure in practice against all currently known machine learning techniques [31]. Nevertheless, this type of PUF would still allow the above attack in 144 min.

Our attacks therefore enforce the use of PUFs with a challenge bitlength of 128 bits or more in Protocols 1 and 2. Since much research currently focuses on 64-bit implementations of electrical PUFs, publication and dissemination of the attack seems important to avoid their use in Protocols 1 and 2. Another aspect of our attack is that it motivates the search for OT and BC protocols that are immune, and which can safely be used with 64-bit implementations. The reason is that the usage of 128-bit PUFs dou-

bles the area consumption of the PUF and negatively affects costs.

4.2 Optical PUFs

Let us now discuss the practical effect of our attack on the optical PUF introduced by Pappu [23] and Pappu et al. [24]. The authors use a cuboid-shaped plastic token of size 1 cm ×1 cm ×2.5 mm, in which thousands of light scattering small spheres are distributed randomly. They analyze the number of applicable, decorrelated challenge–response pairs in their set-up, arriving at a figure of 2.37×10^{10} [24]. Brzuska et al. assume that these challenges are encoded in a set of the form $\{0,1\}^{\lambda}$, in which case $\lambda = \lceil \log_2 2.37 \times 10^{10} \rceil = 35$. If this number of 2^{35} is reduced quadratically by virtue of Lemma 3, we obtain on the order of $2 \times 2^{18} = 5.2 \times 10^5$ CRPs that must be read out by an adversary to cheat. It is clear that even dedicated measurement set-ups for optical PUFs cannot realize the MHz rates of the electrical example in the last section. But even assuming mild read-out rates of 10 or 100 CRPs per second, we still arrive at small read-out times of 5.2×10^4 or 5.2×10^3 s, respectively. This is between 14.4 h (for 10 CRPs per second) or 87 min (for 100 CRPs per second). If a malicious receiver holds the PUF for such a time frame before the protocol starts (which is impossible to control or prevent for the honest players), he can break the protocol's security.

Can the situation be cleared by simply scaling the optical PUF to larger sizes? Unfortunately, also an asymptotic analysis of the situation shows the same picture. All variable parameters of the optical PUF [17,23,24] are the x–y-coordinate of the incident laser beam and the spatial angle Θ under which the laser hits the token. This leads to a merely cubic complexity in the three-dimensional diameter d of the cuboid scattering token.[2] Given our attack, this implies that the adversary must only read out $O(d^{1.5})$ challenges to cheat in Protocols 1 and 2. If only the independent challenges are considered, the picture is yet more drastic: as shown in [37], the PUF has at most a quadratic number of *independent* challenges in d. This reduces to a merely *linear* number of CRPs which the adversary must read out in our attack. Finally, we remark that scaling up the size of the PUF also quickly reaches its limits under practical aspects: the token considered by Pappu et al. [23,24] has an area of 1 cm × 1 cm. To slow down the quadratic attack merely by a factor of 10, a token of area 10 cm × 10 cm would have to be used. Such a token is too large to even fit onto a smart card.

[2] Please note in this context that the claim of [3] that the number of CRPs of an optical PUF is super-polynomial must have been made erroneously or by mistake; our above brief analysis shows that it is at mostly cubic. The low-degree polynomial amount of challenges of the optical PUF is indeed confirmed by the entire literature on the topic, most prominently [23,24,37].

Overall, this leads to the conclusion that optical PUFs like the ones discussed in [17,23,24] cannot be used safely with the Protocols 1 and 2 in the face of our attack. Due to their low-degree polynomial CRP complexity, and due to practical size constraints, simple scaling of the PUFs constitutes no efficient countermeasure. This distinguishes the optical approach from the electrical case of the last section. This observation has a particular relevance, since Brzuska et al. had explicitly suggested optical PUFs for the implementation of their protocols (see Sect. 8 of [3]).

5 Applicability to an OT-protocol of Ostrovsky et al.

Ostrovsky et al. re-examined the use of PUFs in the UC-framework in [20], introducing several new protocols. One of these is an unconditional (i.e., no other assumptions than the security of the PUF are made) OT-procotol for so-called "honest" [20] or "good" [7] PUFs, which is described in Fig. 3 of their paper [20]. We provide it as Protocol 9 in Appendix 8.

Protocol 9 differs from Brzuska et al.'s Protocol 1 in a number of steps. The main reason is that it aims for security in a different UC-simulator model, the so-called oblivious query model. To achieve this new type of security, Ostrovsky et al. use several "random" OT protocols as "subroutines" in their scheme, which are combined in the end to achieve one secure OT protocol.

Still, it can be observed that all of these "random" OT protocols use the same basic OT mechnism as Brzuska et al., and employ the very same PUF $\mathsf{sid}^{\mathsf{R}}$. This makes our previous attack applicable to Ostrovsky et al.'s protocol, too. Due to the exact construction of the protocol, the details are slightly more involved, as we will see below.

The attack. As in Sect. 3, we assume without loss of generality that the PUF has got a challenge set of $\{0,1\}^{\lambda}$ (compare [2,3,20]). Given knowledge of Lemma 3, the receiver can cheat with a quadratic advantage in Protocol 9 as follows.

As in Sect. 3, he uniformly chooses random linear subspaces A and B and constructs the set M, as in Lemma 3. Next, he reads out the responses of $\mathsf{sid}^{\mathsf{R}}$ to all challenges in M. As before, this constitutes a quadratic improvement over reading out the responses to all possible challenges.[3]

Now, when the sender in Step 3(a) of the protocol transfers the value pairs x_i^0, x_i^1 for $1 \leq i \leq 2k$, the receiver computes

[3] In practice, the receiver could either gain sufficient time for this read-out by delaying the protocol in Step 2. Please note in this context that it is part of the UC-model that malicious parties and adversaries can delay messages and the protocol arbitrarily, and that there are no time bounds on the protocol. A practically viable alternative is that the receiver produces the PUF $\mathsf{sid}^{\mathsf{R}}$ already prior to the start of the protocol, giving him sufficient time to apply all challenges in M. The latter approach is exactly equivalent to the situation in Protocol 1.

for each $1 \leq i \leq 2k$ two challenges $e_i^0, e_i^1 \in M$ such that

$$x_i^0 \oplus x_i^1 = e_i^0 \oplus e_i^1.$$

It was shown in Lemma 3 that he can do so efficiently. Next, the receiver deviates maliciously from the protocol: in Step 3(b), for $1 \leq j \leq 2k$, he sets the values

$$v_i := x_i^{b_i} \oplus e_i^0.$$

This choice of the v_i implies that the values $\hat{q}_{i_j}^{\delta}$, which the sender computes in the second item of Step 5, for all $\delta \in \{0, 1\}$ and $1 \leq j \leq k$ will take the following form:

$$\hat{q}_{i_j}^{\delta} = v_{i_j} \oplus x_{i_j}^{\delta} = x_{i_j}^{b_i} \oplus e_{i_j}^0 \oplus x_{i_j}^{\delta} = \begin{cases} e_{i_j}^0, & \text{for } \delta = b_{i_j} \\ e_{i_j}^1, & \text{for } \delta \neq b_{i_j} \end{cases}$$

In particular, all values $\hat{q}_{i_j}^{\delta}$ lie in M. Therefore the receiver knows all responses of $\mathsf{sid}^{\mathsf{R}}$ to the queries $\hat{q}_{i_j}^{\delta}$. Let us call these responses $r_{i_j}^{\delta}$.

The fact that he knows all $r_{i_j}^{\delta}$ now allows the receiver to complete his malicious strategy in Step 6 of the protocol. He sets

$$st_{i_j} \leftarrow \mathsf{FuzRep}(p_{i_j}^{b_{i_j}}, r_{i_j}^{b_i})$$

and

$$st'_{i_j} \leftarrow \mathsf{FuzRep}(p_{i_j}^{1 \oplus b_{i_j}}, r_{i_j}^{1 \oplus b_i}),$$

and recovers both strings s^0, s^1 by computing

$$s^b = \bigoplus_{j=1}^{k} (m_{i_j}^b \oplus st_{i_j}) \quad \text{and} \quad s^{1 \oplus b} = \bigoplus_{j=1}^{k} (m_{i_j}^{1 \oplus b} \oplus st'_{i_j}),$$

thus breaking the security of the protocol.

The presented attack is again simple and effective: it fully works within the original communication model of Ostrovsky et al. [20]. Furthermore, it does not require laborious computations of several days on the side of the attacker (as certain modeling attacks on PUFs do [31]). Finally, due to the special construction we proposed, the honest players will not notice the special choice of the value v, as the latter shows no difference from a randomly chosen value.

Practical consequences. Even though the above attack differs in its details, it has exactly the same practical consequences as the previous attack of Sect. 3, and the discussion of Sect. 4 applies in a completely analog manner.

In a nutshell, Protocol 9 becomes insecure if Pappu's optical PUF [23,24] is used. Provided that the receiver is prepared to invest read-out times of 14.4 h (at read-out speeds of 10 CRPs per second) or of 87 min (at read-out speeds of 100 CRPs per second), he can break the protocol with probability one. Regarding the scaling of the optical PUF, the same arguments as in Sect. 4.2 apply: scaling makes little sense from a practical perspective, since the optical PUF will quickly grow very large for any type of practical applications like smart cards. It is has a limited impact from an asymptotic perspective, also, since the number of CRPs grows only low-degree polynomial in the size of the PUF.

Furthermore, Protocol 9 becomes insecure if XOR Arbiter PUFs or comparable electrical PUFs of bitlength 64 are employed: A read-out time of 144 min at the customary CRP read-out frequency of 1 MHz then suffices to break the protocol. The attack is regardless of the cryptographic hardness of the PUF, or of its resilience against machine learning attacks [31], and only relates to the size of its challenge space. The details are again similar to our discussion in Sect. 4.1.

To restore the security of Protocol 9, the use of an electrical PUF with challenge length 128 or larger seems necessary. This doubles PUF area and costs, and motivates the search for OT and BC protocols that can be used with 64-bit electrical PUFs.

6 A potential countermeasure: additional PUF transfers and time constraints?

Can we bind the time in which the malicious player has got access to the PUF to prevent our attack? The current Protocols 1 and 2 obviously are unsuited to this end; but could there be modifications of theirs which have this property? A simple approach seems the introduction of one additional PUF transfer from the sender to the receiver in the initialization phase. This assumes that the sender initially holds the PUF, transfers it to the receiver, and measures the time period within which the receiver returns the PUF. The (bounded) period in which the receiver had access to the PUF can then be used to derive a bound on the number of CRPs the receiver might know. This could be used to enforce security against a cheating receiver. Please note that a long, uncontrolled access time for the sender is no problem for the protocol's security, whence it suffices to concentrate on the receiver.

On closer inspection, however, there are significant problems with this approach. , which prevent its practical application. In general, each PUF transfer in a protocol is very costly. If executed via physical delivery over long distances between arbitrary parties, it might cost days. One PUF transfer per protocol seems acceptable, since it is often executed automatically and for free, for example by consumers carrying their bank cards to cash machines. But having two such transfers in one protocol, as suggested above, will most often ruin a protocol's practicality.

A second issue is that binding the adversarial access time *in a tight manner* by two consecutive PUF transfers is very difficult. How long will one physical transfer of the PUF take? 1 day? If the adversary can execute this transfer a few

hours faster and can use the gained time for executing measurements on the PUF, our countermeasure fails. The same holds if the adversary carries out the physical transfer himself and can measure the PUF while it is in transit.

In summary, enforcing a tight time bound on the receiver's access time by two PUF transfers or also by other measures will be impossible in almost any applications. The above idea may thus be interesting as a theoretical concept for future PUF protocol design, but cannot be considered a general, practically relevant countermeasure.

7 New protocols based on interactive hashing

Let us now discuss a second and more effective countermeasure: the employment of interactive hashing (IH) as a substep in OT protocols. As we will show, protocols based on IH can achieve better security properties than Protocol 1. The idea of using IH in the context of PUFs has been suggested first by Rührmair in [26]. The following approach is a simplified version of his original scheme. We also give (for the first time) a security analysis of the protocol. Via the general reduction of BC to OT presented in Protocol 2, our construction for OT can also be used to implement PUF-based BC.

7.1 Interactive hashing as a security primitive

Interactive hashing (IH) is a two-player security primitive suggested by [19,22]. It has been deployed as a protocol tool in various contexts, including zero-knowledge proofs, bit commitment and oblivious transfer (see references in [19]. The following easily accessible and application-independent definition of IH has been given in [5]; for more a formal treatment see [33].

Definition 4 (*Interactive hashing (IH)* [5]) Interactive hashing is a cryptographic primitive between two players, the sender and the receiver. It takes as input a string $c \in \{0,1\}^t$ from the sender, and produces as output two t-bit strings, one of which is c and the other $c' \neq c$. The output strings are available to both the sender and the receiver, and satisfy the following properties:

1. The receiver cannot tell which of the two output strings was the original input. Let the two output strings be c_0, c_1, labeled according to lexicographic order. Then if both strings were a priori equally likely to have been the sender's input c, then they are a posteriori equally likely as well.
2. When both participants are honest, the input is equally likely to be paired with any of the other strings. Let c be the sender's input and let c' be the second output of interactive hashing. Then provided that both participants follow the protocol, c' will be uniformly distributed among all $2^t - 1$ strings different from c.
3. The sender cannot force both outputs to have a rare property. Let $\mathcal{G}$ be a subset of $\{0,1\}^t$ representing the sender's "good set". Let G be the cardinality of $\mathcal{G}$ and let $T = 2^t$. Then if G/T is small, the probability that a dishonest sender will succeed in having both outputs c_0, c_1 be in $\mathcal{G}$ is comparably "small".

One standard method to implement IH is by virtue of a classical technique by Naor et al. [19]. To achieve a self-contained treatment, we describe this technique in a variant introduced by Crepeau et al. [5] below. In the protocol below, let c be a t-bit string that is the input to sender in the interactive hashing. All operations take place in the binary field $\mathcal{F}_2$.

Protocol 5 Interactive hashing [5]

1. The receiver chooses a $(t-1) \times t$ matrix $\mathbf{Q}$ uniformly at random among all binary matrices of rank $t-1$. Let q_i be the i-th query, consisting of the i-th row of $\mathbf{Q}$.
2. For $1 \leq i \leq t-1$ do:
   (a) The receiver sends query q_i to the sender.
   (b) The sender responds with $v_i = q_i \cdot c$.
   (c) Given $\mathbf{Q}$ and $v \in \{0,1\}^{t-1}$ (the vector of the sender's responses), both parties compute the two values of $c \in \{0,1\}^t$ consistent with the linear system $\mathbf{Q} \cdot c = v$. These solutions are labeled c_0, c_1 according to lexicographic order.

The following theorem, which is taken from [5,33], tells us about the security of the above scheme. It relates to the security Definition 4.

Theorem 6 (Security of Protocol 5) *Protocol 5 satisfies all three information theoretic security properties of Definition 4. Specifically, for Property 3 of Definition 4, it ensures that a dishonest sender can succeed in causing both outputs to be in the "good set" $\mathcal{G}$ with probability at most* $15.6805 \times G/T$, *where* $G = |\mathcal{G}|$ *and* $T = 2^t$.

Protocol 5 uses t rounds of interaction. Note that interactive hashing is unconditionally secure in the sense that it does not require additional set-up or computational assumptions.

7.2 Oblivious transfer protocol

We now present a PUF-based oblivious transfer protocol that uses IH as a substep. It bears some similarities with an earlier protocol of Rührmair [26] in the sense that it also uses interactive hashing, but is slightly simpler.

Protocol 7 PUF-based 1-out-of-2 oblivious transfer with interactive hashing

1. The sender's input are two strings $s_0, s_1 \in \{0,1\}^\lambda$ and the receiver's input is a bit $b \in \{0,1\}$.
2. The receiver chooses a challenge $c \in \{0,1\}^\lambda$ uniformly at random. He applies c to the PUF, which responds r. He transfers the PUF to the sender.
3. The sender and receiver execute an IH protocol, where the receiver has input c. Both get outputs c_0, c_1. Let i be the value where $c_i = c$.
4. The receiver sends $b' := b \oplus i$ to the sender.
5. The sender applies the challenges c_0 and c_1 to the PUF. Denote the corresponding responses as r_0 and r_1.
6. The sender sends $S_0 := s_0 \oplus r_{b'}$ and $S_1 := s_1 \oplus r_{1-b'}$ to receiver.
7. The receiver recovers the string s_b that depends on his choice bit b as $S_b \oplus r = s_b \oplus r_{b \oplus b'} \oplus r = s_b \oplus r_i \oplus r = s_b$.

7.3 Security and practicality analysis

We start by a security analysis of Protocol 7 in the so-called "stand alone, good PUF model", which was introduced by van Dijk and Rührmair in [7]. In this communication model, the following two assumptions are made: (i) the PUF protocol is executed only once, and the adversary or malicious players have no access to the PUF anymore after the end of the protocol; (ii) the two players do not manipulate the used PUFs on a hardware level. We stress that whenever these two features cannot be guaranteed in practical applications, a number of unexpected attacks apply, which spoil the security of the respective protocols. Even certain impossibility results can be shown under these circumstances; see [7] for details.

In the following analysis in the stand alone, good PUF model, we assume that the adversary has the following capabilities:

1. He knows a certain number of CRPs of the PUF, and has possibly used them to build an (incomplete) predictive model of the PUF. To model this ability, we assume that there is a proper subset $S \subsetneq C$ of the set of all challenges C such that the adversary knows the correct responses to the challenges in S with probability one. The cardinality of S depends on the previous access times of the adversary to the PUF and the number of CRPs he has collected from other sources, for example protocol eavesdropping. It must be estimated by the honest protocol users based on the given application scenario. Usually $|S| \ll |C|$.
2. Furthermore, we assume that the adversary can correctly guess the response to a uniformly and randomly chosen challenge $c \in C \backslash S$ with probability at most ϵ, where the probability is taken over the choice of c and over the adversary's random coins. Usually ϵ will be significantly smaller than one. To name two examples: in the case of a well-designed electrical PUF with single-bit output, ϵ will be around 0.5; in the case of a well-designed optical PUF [23,24] with multi-bit images as outputs, ϵ can be extremely small, for example smaller than 2^{-100}. Again, the honest protocol users must estimate ϵ based on the circumstances and the employed PUF.

Assuming the above capabilities and using Theorem 6, the probability that the receiver can cheat in Protocol 7 is bounded above by

$$15.6805 \times |S|/|C| + \epsilon,$$

a term that will usually be significantly smaller than one.

Under the presumption that this cheating probability of the receiver is indeed smaller than one, the security of Protocol 7 can be further amplified by using a well-known result by Damgard et al. (see Lemma 3 of [6]):

Theorem 8 (OT-amplification [6]) *Let* (p,q)-WOT *be a 1-2-OT protocol where the sender with probability* p *learns the choice bit* c *and the receiver with probability* q *learns the other bit* b_{1-c}. *Assume that* $p+q < 1$. *Then the probabilities* p *and* q *can be reduced by running* k (p,q)-WOT *protocols to obtain a* $(1-(1-p)^k, q^k)$-WOT *protocol.*

In the case of our OT protocol 7 it holds that $p = 0$, whence the technique of Damgard et al. leads to an efficient security amplification, and to a $(0, q^k)$-WOT protocol. The PUF does not need to be transferred k times, but one PUF transfer suffices. We remark that the probability amplification according to Theorem 8 is not possible with Protocol 1 after our quadratic attack, since the attack leads to a cheating probability of one for the receiver, i.e., $p + q \geq 1$ in the language of Theorem 8.

Let us quantatively illustrate the security gain of Protocol 7 over Protocol 1 via a simplified back-of-the-envelope calculation: we argued earlier that via our quadratic attack, a malicious receiver who has read out 2×2^{18} CRPs from an optical PUF can cheat with probability 1 (=with certainty) in Protocol 1. Let us compare this to the case that an optical PUF is used in the IH-based Protocol 7. Let us assume that the adversary has collected the same number of CRPs ($=2 \times 2^{18}$ CRPs) as in the quadratic attack, and that the (multi-bit) response of the optical PUF on the remaining CRPs is still hard to preduct, i.e., it cannot be predicted better than with probability $\epsilon \leq 2^{-100}$. Then by Theorem 6 and by our above analysis, the adversary's chances to break Protocol 7 are merely around $15.6805 \times 2^{19} \times 2^{-35} + 2^{-100} \approx 0.00024$. This probability can then be exponentially reduced further via Theorem 8.

On the downside, however, the IH Protocol 5 has a round complexity that is linear (i.e., equal to $\lambda - 1$) in the security

parameter λ. This is relatively significant for the optical PUF (where $\lambda = 35$) and electrical PUFs with medium bitlengths (where $\lambda = 64$). One possible way to get around this problem is to use the constant round interactive hashing scheme by Ding et al. [8]. However, this scheme has slightly worse security guarantees than the IH scheme of Theorem 6.

To summarize the discussion in this subsection, interactive hashing can restore the security of PUF-based OT protocols even for small-sized PUFs with 64-bit challenge lengths and for optical PUFs in the stand alone, good PUF model. Via the general reduction of BC to OT given in Protocol 2, this result can be used to securely implement PUF-based BC in this model, too.

However, the use of IH leads to an increased number of communication rounds that is about equal to the (binary) challenge length of the PUF, i.e., around 64 rounds for the integrated PUFs with 64 bit challenges, and around 35 rounds for optical PUFs of size 1 cm^2 [24]. It must be decided on the basis of the concrete application scenario whether such a number of rounds is acceptable. If not, then a variant of interactive hashing introduced by Ding et al. [8] can easily be employed, which only involves a constant number (to be precise: four) rounds of communication.

8 Summary and conclusions

We revisited PUF-based OT and BC protocols, including the recent schemes of Rührmair from Trust 2010 [26], Brzuska et al. from Crypto 2011 [2,3], and Ostrovsky et al. [20]. We placed special emphasis on the security which these protocols achieve in practice, in particular when they are used in connection with the well-known optical PUFs and 64-bit electrical PUFs. Our analysis revealed several interesting facts.

First of all, we described a simple and efficient method by which the OT and BC protocol of Brzuska et al. and the OT protocol by Ostrovsky et al. can be attacked with probability one in practice if electrical PUFs with 64-bit challenge lengths are used, or whenever optical PUFs are employed. Since much research focuses on 64-bit implementations of electrical PUFs [10,11,16], and since Brzuska et al. had explicitly suggested optical PUFs for the implementation of their protocols (see Sect. 8 of [3]), the publication and dissemination of our quadratic attack seems important to avoid their use in Protocols 1, 2 and 9. Please note that our attack is independent of the cryptographic hardness of the PUF, and is merely based on its challenge size.

Secondly, we discussed an alternative class of protocols for oblivious transfer that are based on interactive hashing techniques. They are inspired by an earlier OT protocol of Rührmair [26]. We argued that these protocols lead to better security in practice. They can be used safely with 64-bit electrical PUFs. When used with optical PUFs, they lead to better security than the protocols of Brzuska et al., but the security margins are tighter than in the 64-bit case. In both cases, a well-known result by Damgard, Kilian and Savail [6] can be used to reduce the cheating probabilities exponentially.

Our discussion once more shows that PUFs are quite special cryptographic and security tools. Due to their finite nature, asymptotic constants that might usually be hidden in $O(\cdot)$- and $\Theta(\cdot)$-notations become relevant in practice and should be discussed explicitly. Furthermore, their specific nature often allows new and unexpected forms of attacks. One of the aims of our work is to bridge the gap between PUFs in theory and applications; reconciling these two fields seems a necessary prerequisite for a healthy long-term development of the field. We hope that the general methods and the approach of this paper can contribute to this goal.

Recommendations for protocols use and future work. Let us conclude the paper with a condensed recommendation for the practical implementation of PUF-based OT and BC protocols, and by a discussion of future work. First, it is clear from our results that the protocol of Brzuska et al. cannot be used safely with optical PUFs a la Pappu (i.e., with non-integrated optical PUFs of practically reasonable sizes), or with electrical PUFs with challenge lengths around 64 bits.

Secondly, we showed that protocols based on interactive hashing (IH) can achieve better security. These protocols can be employed safely with optical PUFs and with electrical PUFs of challenge length 64. Furthermore, Damgard et al.'s [6] amplification technique can be applied to bring the cheating probabilities arbitrarily close to zero.

Nevertheless, we would like to stress once more to practical PUF users that this analysis only applies if the protocols are employed in the stand alone, good PUF model (see Sect. 7.3 and [7]). As soon as the features of this model cannot be enforced in a given application (for example by certifying a PUF, or by erasing PUF responses at the protocol end, see [7]), certain new attacks apply. They break the security of the presented IH-based protocol, as well as of the protocols of Brzuska et al. and of Ostrovsky et al. that we discussed in this publication. These attacks are not the topic of this work, but have been described in all detail in [7].

Finally, if a PUF has challenge length of 128 bits or more, it seems that the protocols of Brzuska et al. and Ostrovsky et al. can be used safely (as long as we remain in the *stand alone, good PUF model*, see above), even though we did not prove this fact formally in this paper. Leading such a formal proof constitutes an interesting topic for future research.

Acknowledgments The authors would like to thank Stefan Wolf and Jürg Wullschleger for interesting discussions, and Stefan Wolf for suggesting the example in Section 3, page 9 to us. Furthermore, we would like to thank Rafail Ostrovsky, Alessandra Scafuro, Ivan Visconti and Akshai Wadia for illustrating discussions on their OT-protocol. Part of this work was conducted within the physical cryptography project at the TU München.

Appendix: Unconditional oblivious transfer protocol for honest PUFs by Ostrovsky et al.

We give below an OT protocol for honest PUFs by Ostrovsky et al. (see Fig. 3 of [20]). It is "unconditional" in the sense that it only relies on PUFs, i.e., it does not use any other assumptions besides PUFs. In agreement and correspondence with the authors, a few minor typos have been discussed and removed [21]. The most relevant of these typos for our analysis is that in item 2 of Step 5, the PUF that is queried must be $\mathsf{sid}^{\mathsf{R}}$, and not "$\mathsf{sid}_{\mathsf{i_j}}$", as in [20] (note that $\mathsf{sid}_{\mathsf{i_j}}$ is not defined in the protocol, which only uses the PUFs $\mathsf{sid}^{\mathsf{S}}_{\mathsf{i_j}}$ and $\mathsf{sid}^{\mathsf{R}}$).

Protocol 9 Unconditional OT with honest PUFs [20]

Sender's input: Strings $s^0, s^1 \in \{0,1\}^n$.
Receiver's input: Bit $b \in \{0,1\}$.

1. **[($\mathsf{S}_{\mathsf{uncOT}} \Rightarrow \mathsf{R}_{\mathsf{uncOT}}$): Sender PUF initialization]** S initializes $2k$ PUFs $\mathsf{sid}^{\mathsf{S}}_1, \ldots, \mathsf{sid}^{\mathsf{S}}_{2k}$ and sends them R.
2. **[($\mathsf{S}_{\mathsf{uncOT}} \Leftarrow \mathsf{R}_{\mathsf{uncOT}}$): Receiver PUF initialization]** R initializes a PUF $\mathsf{sid}^{\mathsf{R}}$. It uniformly chooses $2k$ queries $q_1, \ldots, q_{2k}$, and obtains responses $a_1, \ldots, a_{2k}$. It sends the PUF $\mathsf{sid}^{\mathsf{R}}$ to S.
3. **[Cut-and-choose]**
   (a) ($\mathsf{S}_{\mathsf{uncOT}} \Rightarrow \mathsf{R}_{\mathsf{uncOT}}$) For $1 \le i \le 2k$, sender uniformly selects a pair of challenges (x_i^0, x_i^1) and sends it to R.
   (b) ($\mathsf{S}_{\mathsf{uncOT}} \Leftarrow \mathsf{R}_{\mathsf{uncOT}}$) For each $1 \le i \le 2k$, receiver does the following:
   – select random bit $b_i \in \{0,1\}$.
   – select random query d_i and let da_i be the response of the PUF $\mathsf{sid}^{\mathsf{S}}_i$. Compute $(dst_i, dp_i) \leftarrow \mathsf{FuzGen}(da_i)$. If $\mathsf{Parity}(dst_i) \neq b_i$, repeat this step. Else, continue.
   – compute $v_i := x_i^{b_i} \oplus q_i$.
   For each $1 \le i \le 2k$, receiver sends to sender (v_i, d_i, dp_i).
   (c) ($\mathsf{S}_{\mathsf{uncOT}} \Rightarrow \mathsf{R}_{\mathsf{uncOT}}$) Sender selects a random subset $S \subset [2k]$ of size k and sends it to receiver.
   (d) ($\mathsf{S}_{\mathsf{uncOT}} \Leftarrow \mathsf{R}_{\mathsf{uncOT}}$) For all $j \in S$, receiver sends (q_j, a_j) to sender, and also hands over the PUF $\mathsf{sid}^{\mathsf{S}}_j$ to the sender.
   (e) Sender makes the following checks for each $j \in S$:
   – compute the response of PUF $\mathsf{sid}^{\mathsf{R}}$ on query q_j to obtain a_j^*. If $\mathsf{dis}(\mathsf{a_j}, \mathsf{a_j^*}) > \mathsf{d_{noise}}$, abort.
   – If $v_j \oplus q_j = x_j^0$, set $b_j^* = 0$; if $v_j \oplus q_j = x_j^1$, set $b_j^* = 1$; else abort.
   – query the PUF $\mathsf{sid}^{\mathsf{S}}_j$ with d_j to obtain response da_j^*; if $\mathsf{Parity}(\mathsf{FuzRep}(da_j^*, dp_j)) \neq b_j^*$, abort.
4. **[($\mathsf{S}_{\mathsf{uncOT}} \Leftarrow \mathsf{R}_{\mathsf{uncOT}}$): Receiver sends correction-bits]** Let $i_1, \ldots, i_k$ be the indices *not* in S. For $1 \le j \le k$, the receiver sends to sender the bit $b'_{i_j} = b_{i_j} \oplus b$.
5. **[($\mathsf{S}_{\mathsf{uncOT}} \Rightarrow \mathsf{R}_{\mathsf{uncOT}}$): Sender's final message]** Sender prepares its final message as follows:
   – For $\delta \in \{0,1\}$, choose random strings $s_1^\delta, \ldots, s_k^\delta$ such that $s^\delta = \bigoplus_{j=1}^k s_j^\delta$.
   – For $\delta \in \{0,1\}$, for $1 \le j \le k$, compute $\hat{q}_{i_j}^\delta = v_{i_j} \oplus x_{i_j}^\delta$, and let $(st_{i_j}^\delta, p_{i_j}^\delta)$ be the response of $\mathsf{PUF}\ \mathsf{sid}^{\mathsf{R}}$ to the query $\hat{q}_{i_j}^\delta$.
   – For $\delta \in \{0,1\}$ and $1 \le j \le k$, set $m_{i_j}^\delta = s_j^\delta \oplus st_{i_j}^{b'_{i_j} \oplus \delta}$.
   The sender sends $(m_{i_1}^0, m_{i_1}^1), \ldots, (m_{i_k}^0, m_{i_k}^1)$ and $(p_{i_1}^0, p_{i_1}^1), \ldots, (p_{i_k}^0, p_{i_k}^1)$ to the receiver.
6. **[Receiver's final step]** For $i \le j \le k$, receiver computes $st_{i_j} \leftarrow \mathsf{FuzRep}(p_{i_j}^{b'_{i_j}}, a_{i_j})$. It outputs $s^b = \bigoplus_{j=1}^k (m_{i_j}^b \oplus st_{i_j})$.

References

1. Armknecht, F., Maes, R., Sadeghi, A.-R., Standaert, F.-X., Wachsmann, C.: A formal foundation for the security features of physical functions. In: IEEE Symposium on Security and Privacy, Oakland, pp. 397–412 (2011)
2. Brzuska, C., Fischlin, M., Schröder, H., Katzenbeisser, S.: Physical unclonable functions in the universal composition framework. In: CRYPTO 2011, Santa Barbara (2011)
3. Brzuska, C., Fischlin, M., Schröder, H., Katzenbeisser, S.: Physical unclonable functions in the universal composition framework. Full version of the paper. Available from IACR Cryptology ePrint Archive 2011: 681 (2011). Accessed 28 Feb 2012
4. Canetti, R.: Universally composable security: a new paradigm for cryptographic protocols. FOCS, pp. 136–145 (2001) (Full and updated version available from IACR Cryptology ePrint Archive 2000: 067 (2000))
5. Crépeau, C., Kilian, J., Savvides, G.: Interactive hashing: an information theoretic tool (invited talk). In: Information Theoretic Security, pp. 14–28. LNCS 5155, Springer, Berlin (2008)
6. Damgard, I., Kilian, J., Salvail, L.: On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In: EUROCRYPT, pp. 56–73. Springer, Heidelberg (1999)
7. van Dijk, M., Rührmair, U.: Physical unclonable functions in cryptographic protocols: security proofs and impossibility results. IACR Cryptology ePrint Archive 2012: 228 (2012)
8. Ding, Y.Z., Harnik, D., Rosen, A., Shaltiel, R.R.: Constant-round oblivious transfer in the bounded storage model. J. Cryptol. **20**(2), 165–202 (2007)
9. Gassend, B.P.: Physical random functions. MSc thesis, MIT, Cambridge (2003)
10. Gassend, B., Clarke, D.E., van Dijk, M., Devadas, S.: Silicon physical random functions. In: ACM Conference on Computer and Communications, Security, New York, pp. 148–160 (2002)
11. Gassend, B., Lim, D., Clarke, D., van Dijk, M., Devadas, S.: Identification and authentication of integrated circuits. Concurr. Comput.: Pract. Exp. **16**(11), 1077–1098 (2004)

12. Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P.: FPGA intrinsic PUFs and their use for IP protection. In: CHES 2007, Springer, Heidelberg, pp. 63–80 (2007)
13. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: STOC 1989, ACM Press, New York, pp. 44–61 (1989)
14. Kilian, J.: Founding cryptography on oblivious transfer. In: STOC 1988, ACM Press, New York (1988)
15. Kumar, S.S., Guajardo, J., Maes, R., Schrijen, G.J., Tuyls, P.: The butterfly PUF: protecting IP on every FPGA. In: HOST 2008, pp. 67–70 (2008)
16. Lee, J.W., Lim, D., Gassend, B., Suh, G.E., van Dijk, M., Devadas, S.: A technique to build a secret key in integrated circuits with identification and authentication applications. In: Proceedings of the IEEE VLSI Circuits Symposium (2004)
17. Maes, R., Verbauwhede, I.: Physically unclonable functions: a study on the state of the art and future research directions. In: Naccache, D., Sadeghi, A.-R. (eds.) Section 1: Towards Hardware-Intrinsic Security. Springer, Berlin (2010)
18. Majzoobi, M., Koushanfar, F., Potkonjak, M.: Lightweight secure PUFs. IC-CAD 2008, San Jose, pp. 607–673 (2008)
19. Naor, M., Ostrovsky, R., Venkatesan, R., Yung, M.: Perfect zero-knowledge arguments for NP using any one-way permutation. J. Cryptol. **11**(2), 87–108 (1998) (Preliminary version in CRYPTO '92)
20. Ostrovsky, R., Scafuro, A., Visconti, I., Wadia, A.: Universally composable secure computation with (malicious) physically uncloneable functions. IACR Cryptology ePrint Archive 2012: 143 (2012). Accessed Sept 2012
21. Ostrovsky, R., Scafuro, A., Visconti, I., Wadia, A.: Personal Communication (2012)
22. Ostrovsky, R., Venkatesan, R., Yung, M.: Fair games against an all powerful adversary. In: AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pp. 155–169 (1993) (Preliminary version in SEQUENCES '91)
23. Pappu, R.: Physical one-way functions. PhD thesis, Massachusetts Institute of Technology, Cambridge (2001)
24. Pappu, R., Recht, B., Taylor, J., Gershenfeld, N.: Physical one-way functions. Science **297**, 2026–2030 (2002)
25. Rivest, R.: Illegitimi non carborundum. Invited keynote talk, CRYPTO 2011 (2011)
26. Rührmair, U.: Oblivious transfer based on physical unclonable functions (extended abstract). In: TRUST 2010, Workshop on Secure Hardware, Berlin, 22 June 2010. Lecture Notes in Computer Science, vol. 6101, pp. 430–440. Springer, Heidelberg (2010)
27. Rührmair, U., Busch, H., Katzenbeisser, S.: Strong PUFs: models, constructions and security proofs. In: Sadeghi, A.-R., Tuyls, P. (eds.) Towards Hardware Intrinsic Security: Foundation and Practice. Springer, Heidelberg (2010)
28. Rührmair, U., Devadas, S., Koushanfar, F.: Security based on physical unclonability and disorder. In: Tehranipoor, M., Wang, C. (eds.) Introduction to Hardware Security and Trust. Springer, New York (2011)
29. Rührmair, U., van Dijk, M.: Practical security analysis of PUF-based two-player protocols. CHES 2012 (2012)
30. Rührmair, U., Jaeger, C., Algasiner, M.: An attack on PUF-based session key exchange, and a hardware-based countermeasure: erasable PUFs. In: Financial Cryptography and Data, Security, St. Lucia (2011)
31. Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., Schmidhuber, J.: Modeling attacks on physical unclonable functions. In: ACM Conference on Computer and Communications Security (CCS'10), Chicago (2010)
32. Rührmair, U., Sölter, J., Sehnke, F.: On the foundations of physical unclonable functions. IACR Cryptology ePrint Archive 2009: 277 (2009)
33. Savvides, G.: Interactive hashing and reductions between oblivious transfer variants. PhD thesis, McGill University, Montreal (2007)
34. Suh, G.E., Devadas, S.: Physical unclonable functions for device authentication and secret key generation. In: DAC 2007, New York, pp. 9–14 (2007)
35. Tuyls, P., Schrijen, G.J., Skoric, B., van Geloven, J., Verhaegh, N., Wolters, R.: Read-proof hardware from protective coatings. In: CHES 2006, Springer, Heidelberg, pp. 369–383 (2006)
36. Tuyls, P., Skoric, B.: Strong authentication with physical unclonable functions. In: Petkovic, M., Jonker, W. (eds.) Security. Privacy and Trust in Modern Data Management. Springer, Heidelberg (2007)
37. Tuyls, P., Skoric, B., Stallinga, S., Akkermans, A., Ophey, W.: Information-theoretic security analysis of physical uncloneable functions. In: Financial Cryptography and Data Security 2005, Roseau (2005)