Terrorist Fraud Resistance of Distance Bounding Protocols Employing Physical Unclonable Functions

Stephan Kleber, Rens W. van der Heijden, Henning Kopp, Frank Kargl
Institute of Distributed Systems
Ulm University, Germany
Email: {stephan.kleber, rens.vanderheijden, henning.kopp, frank.kargl}@uni-ulm.de

***Abstract*— Distance bounding protocols (DBPs) are security protocols that aim to limit the maximum possible distance between two partners in a wireless communication. This enables to ensure locality of interaction between two devices. Despite numerous proposed protocols, recent analyses of DBPs have shown the majority of them to be susceptible to attacks. Most prominent among the unsolved security problems of DBPs is terrorist fraud. This type of attack involves collaboration with a legitimate device, after which the attacker can successfully execute the protocol. We show how terrorist fraud can be prevented by replacing shared secrets – commonly used in classical DBPs – with physical unclonable functions (PUFs). Our new approach can be integrated in all current DBPs with minor modifications. We offer two alternate designs: One utilizing challenge-response PUFs and another using so-called SIMPL systems, a PUF-analogue to public-key cryptography. We use a security model proposed by previous work to demonstrate security of our scheme.**

I. Introduction

In many scenarios, the limited range of wireless communication is used as an authentication feature. There are applications that unlock or lock your computer when a specific Bluetooth device enters or leaves communication range. Similarly, luxury cars often provide keyless entry systems that use a wireless keycard to unlock the car once it is in close proximity [1]. In these scenarios, an inherent assumption is that the very existence of communication is equivalent to proximity of the device and thus proximity of the user.

Francillon et al. [1] identified that this assumption can be circumvented by various forms of relay attacks, and they were able to unlock cars by simple relaying of communication between a distant keycard and the car. The proposed mitigation to such an attack is to integrate a distance bounding protocol (DBP) into the system design. The intuition is to use the inevitable propagation delay of transmitted signals to prove that a device is closer than a specified maximum range.

In a DBP, an authenticator – called "verifier" – engages in a protocol with a remote entity – called "prover". The prover needs to attest to the verifier that it is closer than a certain maximum distance. In recent years, a number of such distance bounding protocols have been proposed [2]–[4].

However, recent work [5] has also shown that several protocols are susceptible to attacks that undermine certain aspects of the protocols' objective. Such attacks, referred to as frauds in the DBP context, trick verifiers to assume that an attacker or another third-party entity is closer to the verifier than it actually is. Some frauds require attackers to be close to the verifier or to collaborate with the legitimate prover.

Fischlin & Onete [5] identified that one of these attacks, terrorist fraud, is especially hard to prevent. They show that many DBPs are vulnerable to this attack. In terrorist fraud, attacker and prover collaborate to establish that the out-of-range prover is at the position of the attacker, within the distance threshold to the verifier. There are deviating definitions of terrorist fraud, which allow different extents of collaboration: The worst case is that the prover shares all its information with the attacker, except for a shared secret between verifier and prover. In case of a terrorist-fraud-susceptible protocol, the attacker hereby gains enough knowledge to successfully deceive the verifier.

In this paper, we propose a generic enhancement to DBPs by using physical unclonable functions (PUFs) to effectively prevent terrorist fraud. Our enhancement relies on the impossibility of cloning and disclosing the PUF, which renders collaboration with the attacker impossible. This even prevents a stronger variant form of terrorist fraud where the attacker is allowed to copy all the prover's data including shared secrets.

In the following Section II, we first introduce the concepts of DBPs and PUFs in detail. This is followed by a discussion of the security characteristics and the attacker model of DBP protocols in Section III. Section IV introduces our basic PUF-based DBP scheme (PUF-DBP) and the use of an alternative to classical PUFs, so-called SIMPL systems. This is followed by a detailed security discussion in Section V. Section VI discusses further implementation details and enhancements to the PUF-DBP concept. Finally, we conclude the paper with an outlook to our future research in Section VII.

II. Background

A. Distance Bounding Protocols

Distance bounding protocols (DBPs) were originally proposed by Brands & Chaum [6] to enable an authenticator to verify the distance to another entity. In this setting, the authenticator is called a "verifier" $\mathcal{V}$ and the entity is called "prover" $\mathcal{P}$. For measuring this distance, a radio signal is

2015 International Conference and Workshops on Networked Systems | DOI: 10.1109/NetSys.2015.7089068

transmitted by $\mathcal{V}$ to $\mathcal{P}$. $\mathcal{P}$ responds by sending back a radio signal. Then $\mathcal{V}$ can measure the round trip time (RTT) between sending its signal and receiving $\mathcal{P}$'s signal. The intuition of this generic distance bounding principle can be seen in Figure 1.

DBPs make use of the inevitable physical boundary of the speed of light c to allow $\mathcal{V}$ to attest that $\mathcal{P}$ is within a certain physical distance. Because no signal can travel faster than the speed of light, the RTT of the signals between $\mathcal{V}$ and $\mathcal{P}$ allows an approximation of the distance as follows: Let t_s be the propagation time of a signal between $\mathcal{V}$ and $\mathcal{P}$. Then, the corresponding distance d_s can be calculated as $d_s = t_s \cdot c$. $\mathcal{V}$ defines a maximum distance d_{max} and a corresponding propagation time t_{max} within which $\mathcal{P}$ is considered valid or in range of $\mathcal{V}$ [7].

1) Measuring distances: The desired RTT is $2t_s$ i.e., the signal propagation time back and forth. However, $\mathcal{P}$ will also need processing time between receiving $\mathcal{V}$'s signal and transmitting its own signal. So the actually measurable RTT includes a processing time t_{proc}. Consequently, the measured-RTT is $2t_s + t_{proc}$. In general, $\mathcal{V}$ cannot make assumptions about t_{proc} for any $\mathcal{P}$. Thus, $\mathcal{V}$ can only resort to an upper bound to measure the physical distance:

$$t_s \cdot c = d_s \leq \frac{(2t_s + t_{proc}) \cdot c}{2}$$

Thus, $\mathcal{P}$ might seem to $\mathcal{V}$ to be farther away than it actually is, making a short processing time of $\mathcal{P}$ desirable.

Apart from processing time, effects like multi-path propagation blur the measurement of a single RTT, an effect that is amplified by the signal length. For precise measurements it is desirable to eliminate these effects as much as possible. Because shorter signals get less distorted and also require less processing time [8], only a single symbol is exchanged when measuring the distance [7].

2) Challenge-Response: A naïve DBP has inherent security problems: the attacker $\mathcal{A}$ can impersonate $\mathcal{P}$ and trick $\mathcal{V}$ to believe $\mathcal{P}$ is nearby. Even when $\mathcal{A}$ is outside $\mathcal{V}$'s valid range, she could send an arbitrary signal before the request from $\mathcal{V}$ reaches her. $\mathcal{V}$ cannot distinguish a genuine signal within the range from the attacker's signal. Therefore, DBPs employ a challenge-response approach, where $\mathcal{V}$ sends a specific challenge and expects $\mathcal{P}$ to answer with an appropriate response.

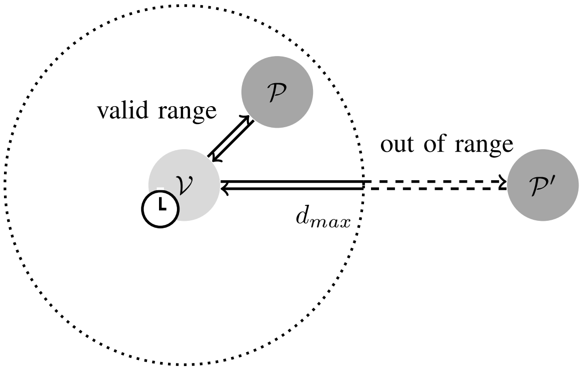

Fig. 1. The generic distance bounding principle.

For simplicity, we adopt the common notion of a DBP's challenge and response to be a binary symbol and call one such symbol "bit" [9], which is just a logical approximation of the real radio signal. Since $\mathcal{V}$ and $\mathcal{P}$ are sending only one bit each, the probability for anyone including an attacker to guess the right response is $\frac{1}{2}$.

To be secure and rule out signal reception errors, typically multiple rounds of a challenge-response dialog between $\mathcal{V}$ and $\mathcal{P}$ are conducted. Each round reduces the probability for an attacker to guess the right response by the factor $\frac{1}{2}$. The necessary number of rounds consequently depends on the desired security level.

3) Typical protocol structure: To exchange initialization data, agree on shared secrets, or verify identities, most DBPs rely on several phases. The whole multiple-RTT measurement process explained above is typically referred to as the *time-critical* phase. All other phases required are subsumed under the term *lazy phases*, with no timing constraints [10]. These may take place before and after the time-critical phase. Most DBPs consist of one lazy *preparation phase* followed by one time-critical phase for the signal RTT measurements. During the preparation phase, arbitrary initialization data may be exchanged, like identity information, nonces, and key agreement. Some protocols, like [6] and [4], define a second lazy phase after the time-critical phase with diverse functions, e. g., refreshing the shared secret.

In many existing DBPs, $\mathcal{P}$ employs a pseudo random function (PRF) as cryptographic primitive. However, PRFs need some sort of shared secret sk as part of their input to be useful in generating a response to a challenge. This leads to the need for a key distribution scheme, typically at the deployment of $\mathcal{P}$. In addition, most DBP schemes use some information from the preparation phase as part of the input to the PRF. This might be just nonces [4], [11] (like $N_{\mathcal{V}}$ in Figure 2) or identity information [3], [12]. From that input, the PRF generates a bit string which is then typically split up into two parts. So, a simplistic usage of a PRF in $\mathcal{P}$ would be: $v^0 \| v^1 = PRF(sk, N_{\mathcal{V}})$. The responses of the time-critical phase here are generated by using the value of the one bit challenge c_i to select either v^0 or v^1 and responding in round i with the corresponding bit $r_i = v_i^{c_i}$ at position i. For an example, see the Hancke and Kuhn [11] protocol in Figure 2.

After a successful DBP run – consisting of a time-critical and lazy phases – $\mathcal{V}$ is convinced that $\mathcal{P}$ is within range d_{max}. Additionally, $\mathcal{V}$ should be able to associate any further phases in this session with $\mathcal{P}$. Having completed a DBP run does not necessarily mean $\mathcal{V}$ has any information about the identity of $\mathcal{P}$: authorization can occur just through the presence of any valid $\mathcal{P}$ within t_{max}. If the application additionally depends on the identity of $\mathcal{P}$, it has to be determined separately. For that, the shared secret required in some DBPs may or may not be sufficient depending on the actual use case. This, however, goes beyond of the scope of the current discussion.

Our approach is intended to counteract terrorist fraud described in Section III employing PUFs and SIMPL systems. Therefore, we next explain these concepts.

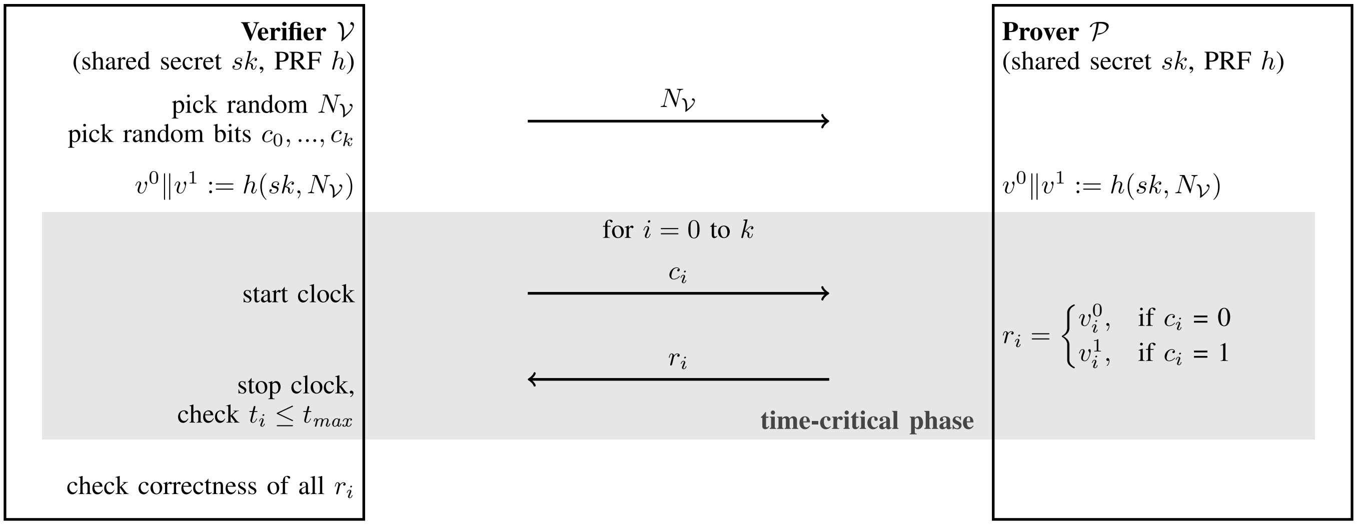
Verifier $\mathcal{V}$
(shared secret sk, PRF h)
pick random $N_\mathcal{V}$
pick random bits $c_0, ..., c_k$
$v^0 \| v^1 := h(sk, N_\mathcal{V})$
start clock
stop clock,
check $t_i \leq t_{max}$
check correctness of all r_i
$N_\mathcal{V}$
for $i = 0$ to k
c_i
r_i
time-critical phase
Prover $\mathcal{P}$
(shared secret sk, PRF h)
$v^0 \| v^1 := h(sk, N_\mathcal{V})$
$r_i = \begin{cases} v_i^0, & \text{if } c_i = 0 \\ v_i^1, & \text{if } c_i = 1 \end{cases}$

Fig. 2. Hancke and Kuhn [11] considered as DBP baseline.

B. Physical Unclonable Functions

The exact definition of physical unclonable functions (PUFs) is subject of ongoing discussion [13]–[15]. According to Rührmair et al. [15], a PUF is a function that maps challenges to responses, one of which is called a *challenge-response-pair* (CRP). This mapping is created by the behavior of a complex physical object that accepts a large number of distinct challenges as input and outputs corresponding responses. An input may be thought of as stimulus to the object. Rührmair et al. call this a *strong t-PUF*, with t being a security parameter. For details and the formal definition refer to [15]. In our work, we refer to these as PUFs. Other types of PUFs exist, such as the *obfuscating t-PUF* [15]. Rather than providing CRPs, this PUF provides secure key storage.

In the following, we define several requirements for a PUF in order to be useful in the context of DBPs. Most importantly, we require the challenge-response property of a strong t-PUF, such that it can replace the PRF in a DBP. The following additional requirements are important for our work:

We assume that a PUF cannot be distinguished from a random oracle. This assumption is somewhat strong [16] although it has been argued before [17], based on a similar argument from Bellare & Rogaway [18].

The characteristics of a PUF, and therefore its challenge-response-mapping, are individual for each instance. Thus, they unambiguously relate a measurable property to a physical object [19]. The characteristics silicon-based PUFs typically exploit are submicron variations in the chip manufacturing process to obtain individual behavior for each instance of the same chip [20]. These characteristics can not be predefined by the manufacturer and can be predicted by an attacker with only negligible probability. Since characteristics can not be predefined, they cannot be cloned by another physical instance. Similarly, since they are unpredictable, no model can compute the outcome within a defined period of time. Rührmair et al. [15] mention a period of one day. This would suffice in our use case, as will become clear in Section IV.

For silicon-based PUFs, the property typically measured constitutes an individual reproducible binary value. Reproducibility means that a response is always the same for the same challenge. Such challenge-response characteristics can be used as a cryptographic primitive [15]. This reproducibility is called robustness in the PUF context [14].

Actually, the properties discussed above do not apply to most PUF designs. Producing the response for a challenge relies on physical measurements of the chip. This inevitably leads to small deviations in the resulting values making them useless for cryptographic applications [19], [20]. To compensate for these deviations, fuzzy extractors are used [14] to ensure reproducibility. Fuzzy extractors are an error correction mechanism typically requiring helper data, generated from readouts of CRPs.

A vulnerability of some PUF designs is that machine learning could be employed to create a model of its characteristics [21] that can be used to predict a PUF's response without involving the physical object. This undermines the postulated unclonability and unpredictability. Proposals to overcome this issue have been made [22], though it is unclear whether those countermeasures are sufficient. Nevertheless, this work shows progress towards machine learning resilience for PUFs, which leads us to assume that the desired properties are feasible. In addition to machine learning, other complex hardware and side-channel attacks on PUFs exist [21], [23]. Techniques from tamper-resistant or -evident hardware could help here, but this is out of scope for our work.

To summarize, we define a PUF to be a physical semiconductor based challenge-response component (CR-PUF), with the properties of robustness, unpredictability and unclonability as stated. An additional requirement is that it is not distinguishable from a random oracle. Finally, we require that the PUF has a sufficiently short runtime, such that it can be usefully deployed in a DBP. Some existing PUFs, like Arbiter or Butterfly PUFs [24], can fulfill this timing requirement, although there are also PUFs that take up to seconds in the worst case [25]. In the end it might be necessary to reach a trade-off between security properties and runtime when choosing a PUF design for implementation.

C. SIMPL Systems

Another recently emerged kind of PUFs, SIMPL (SIMulation Possible but Laborious) systems, are immune to model building attacks, as their model is public. The definition of unclonability is weakened for these systems [26]–[28]: it guarantees that the correct response to a challenge cannot be computed any faster than by the original PUF instance. Moreover, with the model anyone can compute the right response, but requires considerably more time for this computation. More formally, for any model m for a PUF p with processing time t_p for the PUF and t_m for the model, it must hold that $t_m \gg t_p$. In this case, the PUF p is called a *public PUF* (PPUF) or *SIMPL system* with the mentioned weak unclonability property.

This property matches nicely with the requirements for DBPs where the timing of the response is crucial. Assume a correct response can be computed by an attacker by modeling the prover. Even then, the model must deliver the result with almost equal delay compared to the legitimate prover. The higher the delay, the more likely a verifier will not accept the response, since it must assume the verifier is too far away. The model must be at least as slow as the duration of the signal propagation for the maximum allowed distance between prover and verifier ($\frac{t_{max}}{2}$, see Section VI): $t_m \geq t_{max}$. In this case, even an attacker with the correct model and the challenges eavesdropped from the time-critical DBP-phase will not be able to successfully commit distance fraud, assuming the challenges are unpredictable.

D. Related Work

An approach related to the scheme we present using a PUF as part of a distance bounding protocol has been proposed by Kardaş et al. [29]. Their concept makes use of the PUF as an immediate replacement for the PRF. This removes the need for a long term shared secret between $\mathcal{P}$ and $\mathcal{V}$. A comparably complex process is needed to derive pre-challenges during the lazy phase of the protocol. Intermediate results have to be stored temporarily by $\mathcal{P}$, potentially enabling attacks by reading out the memory before the results are erased. Similarly, an attacker with access to the memory of the prover during the lazy phase is able to get the pre-computed response values, similar to conventional DBP schemes. Thus, Kardaş et al. make no special use of a PUF's properties for the mitigation of terrorist fraud.

In contrast, our approach shifts the usage of the PUF closer to the response generation, preventing unencrypted storage of response values and reducing the attack surface at the same time. This way, no attack that only exploits the timing agnostic lazy phases, is feasible. Moreover direct usage of the PUF during time-critical phases considerably reduces the required communication and computation complexity of the protocol, compared to [29]. Therefore, our approach is the first that exploits the full potential of PUFs for DBPs.

III. Attacker Model

Classically, security protocols are tested in the Dolev-Yao model, which provides an intuitive strong attacker (often expressed as *attacker carries the message*), while assuming that the relevant cryptographic primitives are secure. Using this model we can test whether an attacker can obtain information or break authentication without breaking cryptographic primitives. However, Dolev-Yao does not allow us to express timing or proximity. In DBPs, exactly these additional elements are used to provide the expected functionality. To analyze DBPs, Dürholz et al. [30] have developed a formal model that considers attacks that exploit proximity and timing.

To provide the intuition behind the formal definitions of frauds given by [30], consider the following example scenario: Alice works at a banking organization that uses personalized RFID cards as $\mathcal{P}$ to provide access and track the working hours of their employees. Bob is a university employee that is befriended with Alice.

a) Distance Fraud: In distance fraud, the attacker attempts to falsely claim proximity to $\mathcal{V}$. For example, when Alice decides to skip work one day, she may attempt to authenticate from home using her card, so that the system will log that she arrived and left. Protocols secure against distance fraud will deny that Alice was near the reader for this session.

b) Mafia Fraud: Mafia fraud refers to an attack where the attacker, Mallory, attempts to gain access to the bank. In this scenario, Mallory has an accomplice Eve, who communicates with Alice's RFID card, while Mallory attempts to authenticate to the bank's RFID reader. Mallory and Eve act as simple relays between the reader and Alice's card.

c) Terrorist Fraud: In terrorist fraud, a malicious $\mathcal{P}$ is able to temporarily delegate its ability to successfully run a DBP with the corresponding $\mathcal{V}$. The attacker gets temporary help from a collaborating legitimate $\mathcal{P}$. As long as $\mathcal{P}$ chooses to aid the attacker, the latter illegitimately gains the ability to get distance attestation by $\mathcal{P}$. As soon as $\mathcal{P}$ withdraws its help, however, the attacker must retain no advantage over its success probability as it was before receiving aid. This implicitly states that $\mathcal{P}$ may not hand over its shared secret nor parts of it that lead to future advantage of the attacker. Handing over the shared secret must be considered a trivial attack. The model is limited further, in that any help of $\mathcal{P}$ might only be given during lazy phases. This corresponds with the intuition that time-critical phases leave no time to $\mathcal{P}$ for side tasks. In the scenario, Alice might devise a card continuously linked to her own. She hands it over to Bob for him to gain access to the bank. Later attempts of Bob to gain access will fail, if she chooses to sever the link between the cards. In particular, most DBPs are vulnerable to terrorist fraud in that $\mathcal{P}$ can hand over a temporary secret. It typically is derived from the shared key, unique for a session of $\mathcal{P}$ and $\mathcal{V}$, and unpredictable to any single one of the parties through the usage of nonces. After a session is established, however, $\mathcal{P}$ may hand over the temporary secret, without leaking its key, but enabling an attacker to successfully conduct a subsequent time-critical phase with the same $\mathcal{V}$.

Fischlin et al. [5] use the formal model [30] to analyze several protocols from the literature. They claim no existing protocol is secure against all of these frauds, although they do note that two attacks they produce may be considered contrived attacks. Excluding these, only an updated version of the Swiss-Knife protocol [4] (in [30]) is secure against all of the above criteria. We discuss the terrorist fraud attacker in Section V, where we also prove our protocol secure against it. An additional type of attack is the impersonation attack, where the attacker bypasses a part of the protocol; this should be prevented by establishing mutual authentication.

IV. PUF-based Terrorist-Fraud Resistant DBP

A. General Idea

The central aspect of our approach is to replace the PRF in $\mathcal{P}$ of conventional DBPs by a PUF. We first explain our scheme as applicable to both, CR-PUFs and SIMPL systems, subsuming both under the term PUF, notwithstanding their differences in general. After having established the general functionality in Sections IV-B1 and IV-B2 we will differentiate between specifics of CR-PUFs and SIMPL systems in the subsequent sections.

Either way, the need for a common shared secret stored in the $\mathcal{P}$ is eliminated, preventing the sharing of that secret. By replacing the PRF, and thus initialization information, a prover collaborating in terrorist fraud cannot give this information to the attacker. Based on this observation, we argue that the enhancement of any otherwise secure DBP-design with our PUF-driven scheme additionally mitigates the threat of terrorist fraud. In principle, a collaborating $\mathcal{P}$ might provide an attacker with arbitrary challenges and responses of the PUF. Given a sufficiently large challenge-response domain of the PUF, the attacker does not gain any significant advantage by this collaboration. The specific challenge that $\mathcal{V}$ will probe $\mathcal{P}$ with is used just once and subsequently invalidated in $\mathcal{V}$. Therefore, any knowledge of a past DBP run is of no use to the attacker, subsequently. By definition of terrorist fraud and a PUF, the PUF itself cannot be handed over by $\mathcal{P}$.

B. Phases

As usual for DBPs, the actual distance measurement is conducted in the time-critical phase. In addition, we require a preparation phase before the time-critical phase for the basic functionality of our scheme. In the following, we describe each of the phases and their functionality separately (see also Figure 3).

1) Preparation: In the time-critical phase, just one challenge bit can be sent to $\mathcal{P}$ per round. $\mathcal{P}$'s PUF, however, needs a bit string of fixed length n as challenge. Therefore, $\mathcal{V}$ transmits (in the clear) a pre-challenge PC to $\mathcal{P}$ (see Fig. 3). The pre-challenge is used to generate challenges for the PUF during the time-critical phase, in combination with the bit c_i.

$\mathcal{V}$ knows a number of CRPs for each $\mathcal{P}$. There is always a set of CRPs corresponding to PC necessary to complete a DBP run. To prevent replay attacks, a set of corresponding CRPs must be considered invalid as soon as the PC is sent, meaning they will not be re-used in future protocol runs.

2) Time-Critical: The *time-critical phase* consists of n transmission rounds for the RTT measurement. In the i-th round, let $c_i \in \{0,1\}$ be the challenge bit sent by $\mathcal{V}$ and $r_i \in \{0,1\}$ the response sent back by $\mathcal{P}$. Figure 3 depicts this phase marked gray, showing one loop i of n rounds.

In each round, we evaluate the PUF to generate a fresh r_i, as opposed to most other DBP schemes, which select a bit r_i from a precomputed PRF output. However, the PUF needs n bits as challenge, so we introduce PC to fill up the remaining bits. However, PC might be overheard by the attacker during the preparation phase, narrowing down the challenge enough for a modeling attack of the PUF to become feasible. To prevent this, PC is successively replaced completely by a round challenge RC:

All $c_j, j \in [0..i]$ up to the current round i of this phase are concatenated into the i-th round's challenge bit string $RC_{[0..i]}$ in the manner $RC_{[0..i]} = c_0\|c_1\|..\|c_i$. The pre-challenge, known from the preparation phase, then is truncated to length $(n-2)-i$: $PC_{[0..(n-2)-i]}$. Therefore $n-1$ bits are sufficient for PC. Consequently the pre-challenge of length $n-1$ bits consists of bits $i = [0..(n-2)]$. $PC_{[0..(n-2)-i]}$ and $RC_{[0..i]}$ both then are concatenated into the challenge C_i of length n bits in this round:

$$C_i = PC_{[0..(n-2)-i]}\|RC_{[0..i]}$$

To generate r_i, the challenge C_i is fed into the PUF: $r_i = \mathrm{PUF}(C_i)$. Already during the preparation phase, $\mathcal{V}$ selects PC

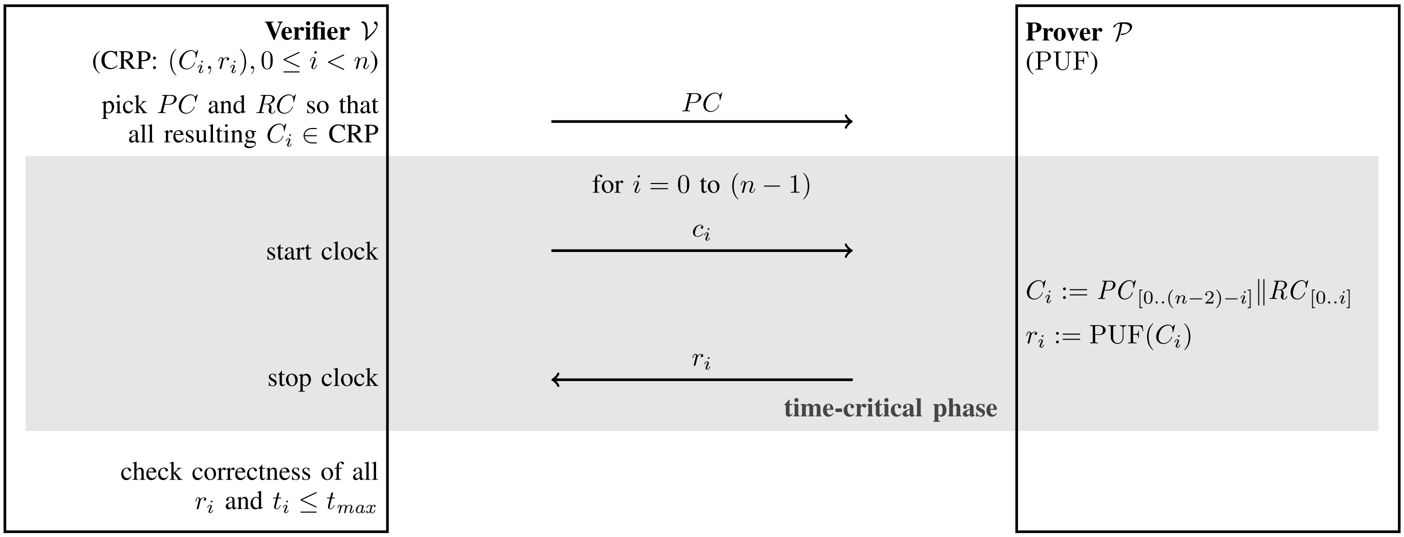
Verifier $\mathcal{V}$
(CRP: $(C_i, r_i), 0 \le i < n$)
pick PC and RC so that all resulting $C_i \in$ CRP
Prover $\mathcal{P}$
(PUF)
PC
for $i = 0$ to $(n-1)$
start clock
c_i
$C_i := PC_{[0..(n-2)-i]}\|RC_{[0..i]}$
$r_i := \mathrm{PUF}(C_i)$
stop clock
r_i
time-critical phase
check correctness of all r_i and $t_i \le t_{max}$

Fig. 3. PUF-DBP protocol run. In this Figure, n is the total amount of RTT-measurements in the one time-critical phase of one protocol run.

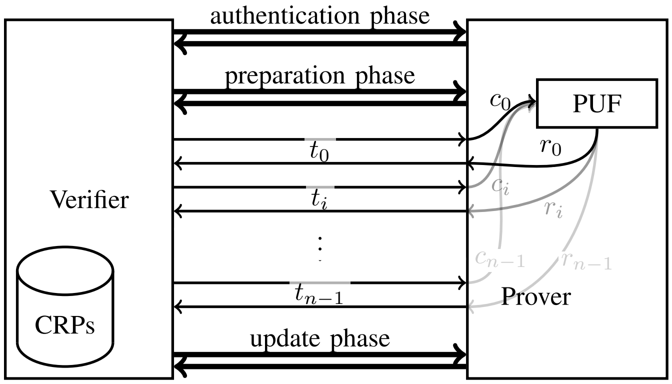

Fig. 4. The challenge-response-PUF DBP principle.

and the set of c_i in such a way that all intermediate C_i that will be generated at $\mathcal{P}$ and result in r_i belong to a CRP (C_i, r_i), each with $0 \leq i < n$, known to $\mathcal{V}$. This means $\mathcal{V}$ has to know at least n of $\mathcal{P}$'s CRPs in advance for one DBP run to complete. These CRPs have to be chosen in such a way that each (C_i, r_i) can be generated by the above algorithm.

3) Authentication and Update: After having introduced the preparation and time-critical phase of our scheme, we now will discuss the subtleties to use either CR-PUFs or SIMPL systems for the implementation.

Considering the properties of **CR-PUFs**, while $\mathcal{V}$ will choose arbitrary challenges, it can not compute the responses without $\mathcal{P}$. Thus, utilizing CR-PUFs, a CRP database and a way to update it is necessary. The idea of a final *update phase*, similar to [3], can be used here. We note that for each $\mathcal{P}$ at least n of its CRPs have to be transmitted after each DBP run to refill $\mathcal{V}$'s CRP database. This might not be feasible in every use case. Thus an initial readout of the necessary CRPs for the whole lifespan of $\mathcal{P}$ might be considered at deployment time.

For a similar reason, CR-PUFs also need the introduction of an *authentication phase* preceding the preparation phase (see Figure 4): $\mathcal{V}$ not being able to calculate new CRPs without the help of $\mathcal{P}$, enables a type of denial of service attack, the *CRP-depletion attack*. Because the CRPs associated with PC must be discarded after transmission of PC to prevent read-out attacks, $\mathcal{V}$ can be depleted from known and valid CRPs. If CR-PUFs are to be employed for a DBP, some well-studied scheme from the literature should be used to provide mutual authentication [31].

Employing **SIMPL systems** for the implementation renders a CRP database unnecessary. $\mathcal{V}$ receives a model of $\mathcal{P}$'s PUF at deployment and can derive an arbitrary number of valid responses from it. However, $\mathcal{V}$ needs to store all models and needs the computing power to calculate them. Although the model per definition has to compute much slower than the physical PUF, temporarily storing $\mathcal{P}$'s responses leaves $\mathcal{V}$ almost indefinite time for this. Since the model does not change and no CRPs have to be refreshed, no update phase is required. Consequently, CRP-depletion is not possible obviating the need for the authentication phase.

V. Security Analysis

In this section we present the security analysis of our protocol improvements. These improvements are not specific to any distance bounding protocol, and as such we will not analyze them in the same way. Instead, we assume a DBP that is secure against distance and mafia fraud, as defined in Section III, and discuss why our improvements provide security against terrorist fraud given this assumption. This is reasonable, because as [30] notes, the three fraud types are independent.

Dürholz et al. [30] developed a model for formally assessing terrorist fraud, which we will discuss here. Let, as before, $\mathcal{V}$ be the verifier, $\mathcal{P}$ be the prover, and n be the number of transmission rounds during the time-critical phase. Let $T_{\max} < n$ be the number of time-critical transmission rounds, which can be completed too late and $E_{\max} < n$ be the number of transmissions where $\mathcal{P}$ can send a wrong response bit to $\mathcal{V}$. DBPs have to include this tolerance to make up for any errors in the network.

We then go on to introduce an attacker $\mathcal{A}$, who is able to access $\mathcal{P}$ to which it can impersonate $\mathcal{V}$ and $\mathcal{V}$ to which it can impersonate $\mathcal{P}$. Furthermore we have a simulator $\mathcal{S}$. The idea is that no simulator $\mathcal{S}$ can use his transcript of the conversation between $\mathcal{P}$ and $\mathcal{V}$ to run the protocol again successfully without the help of $\mathcal{P}$, i. e., when $\mathcal{S}$ is not able to access $\mathcal{P}$. Thus the simulator is a common attacker, whereas $\mathcal{A}$ is a terrorist-fraud attacker with the added advantage of getting help from $\mathcal{P}$. Then we can quantify the advantage obtained through collaboration with $\mathcal{P}$ by

$$\mathrm{Adv}(\mathcal{A}, \mathcal{S}, \mathcal{P}) = p_{\mathcal{A}} - p_{\mathcal{S}}$$

where $p_{\mathcal{A}}$ is the probability that $\mathcal{A}$ successfully runs the protocol with $\mathcal{V}$ such that at most $T_{\max}$ transmission rounds are tainted. For a formal definition of a tainted phase see the original paper [30]. $p_{\mathcal{S}}$ is the probability that $\mathcal{S}$ makes $\mathcal{V}$ accept one of the subsequent sessions, given the same knowledge as $\mathcal{A}$, i. e., $\mathcal{S}$ can communicate with $\mathcal{A}$ in an offline phase. The number of challenges anyone must have correct for a successful run of the protocol assuming a reliable and lossless channel and immediate responses is then $n - E_{\max}$. Therefore $p_{\mathcal{S}} = (\frac{1}{2})^{n-E_{\max}}$ for random guessing. Otherwise it holds that $1 \geq p_{\mathcal{S}} \geq (\frac{1}{2})^{n-E_{\max}}$ when $\mathcal{S}$ has the added knowledge of $\mathcal{A}$.

The number of challenges $\mathcal{A}$ must have correct for a successful run of the protocol, assuming he taints the first $T_{\max}$ transmission rounds, is at most $n - E_{\max} - T_{\max}$. Thus we obtain $1 \geq p_{\mathcal{A}} \geq (\frac{1}{2})^{n-E_{\max}-T_{\max}}$.

Assume now that $p_{\mathcal{A}} > (\frac{1}{2})^{n-E_{\max}-T_{\max}}$. Recall that the response bit r_i in every transmission round depends on the PUF applied to part of the pre-challenge and all previous challenge bits. This would mean that $\mathcal{A}$ is able to model the output of the PUF in less time than $t_{\max}$ for $n - E_{\max} - T_{\max}$ rounds, since by the assumption above he predicts the output better than random guessing. Especially that would imply that the attacker has a distinguisher for the PUF. Recalling that a PUF is assumed to be indistinguishable from a random oracle (see Section II-B), this is a contradiction to the properties of a PUF and thus $p_{\mathcal{A}} = (\frac{1}{2})^{n-E_{\max}-T_{\max}}$ to begin with.

We then obtain

$$\begin{aligned}\mathrm{Adv}(\mathcal{A},\mathcal{S},\mathcal{P}) &\leq \left(\frac{1}{2}\right)^{n-E_{\max}-T_{\max}} - \left(\frac{1}{2}\right)^{n-E_{\max}} \\ &= \frac{2^{T_{\max}}-1}{2^{n-E_{\max}}}\end{aligned}$$

which for sufficiently large $n - E_{\max}$ is negligible. That means that $\mathcal{P}$ can help $\mathcal{A}$ only insignificantly if we choose the tolerance against false responses low enough.

VI. PUF-Specific Enhancement

To overcome practical problems we identified in the initial approach, we propose an enhancement to the basic scheme, we call *Timing Extension*. It exploits the analogue nature of typical PUFs' circuitry.

As explained, DBPs can only guarantee an upper bound on the physical distance between $\mathcal{V}$ and $\mathcal{P}$ since the processing time t_{proc} at $\mathcal{P}$ in general can not be accounted for by $\mathcal{V}$ with certainty. To reduce the uncertainty of distance introduced by t_{proc}, we employ the properties of PUFs that have a *physically guaranteed lower bound* for their processing time t_{proc}, and this t_{proc} is constant for arbitrary different challenges and there is no feasible way of pre-computing or deducing the response to a given challenge with more confidence than $\frac{1}{2}$. Then it is possible to consider t_{proc} in the protocol to narrow down the uncertainty of distance.

For this we define in accordance with the previous definitions: t_p as processing time of the PUF; t_v as overhead of $\mathcal{V}$ to perform the RTT measurement; and let $t_s = \frac{d_s}{c}$ be the time required for the signal propagation over the distance d_s. Furthermore t_d is some arbitrary delay, not accounted for otherwise. Then, the overhead of one round is: $t_{proc} = t_p + t_v + t_d$ while the measurement of a round will provide $t_i = 2t_s + t_{proc}$. Consequently, with lower bounds for t_p and t_v we can deduce a tighter upper bound for the distance between $\mathcal{P}$ and $\mathcal{V}$:

$$d_s \leq c \cdot \frac{t_i - t_p - t_v}{2} \leq c \cdot \frac{t_i}{2}$$

Since the verifier can measure t_i (and will do so in each round), and the verifier knows the values of c and (at least a lower boundary for) t_p and t_v, it can deduce that the prover is no farther away than d_s, regardless of the timing of the PUF and its helper circuits. This allows for PUFs with strict security properties and longer processing time without having to sacrifice distance boundary precision.

VII. Conclusion

We presented a combination of DBPs and PUFs that can be proven to be secure against terrorist fraud in the formal model of Dürholz et al. Our combination can be applied to different DBPs and does not require to limit the terrorist-fraud attacker model. We defined requirements for a PUF that are suitable for this use case, whose properties we then used to build the DBP. In general, PUFs are much faster in generating a response from a challenge than PRFs, and therefore, they can be used during the time-critical phase. We have shown how this can be exploited to prevent a collaborating prover in the context of terrorist fraud.

We considered obstacles a CR-PUF involves for the implementation of a DBP to demonstrate the required extension of the scheme's phases. The drawbacks of such an implementation mainly relate to the property of unclonability of PUFs themselves. We require this property for our protocol to be secure, so that the collaborating prover can not disclose information about the PUF to the attacker. In the course of this considerations, it became apparent, that a SIMPL system instead of a CR-PUF solves most of the problems, given such a system is feasible for a use case. The SIMPL system allows us to avoid the significant memory overhead of a CRP database for each registered prover and prevents CRP-depletion attacks. This renders the additional authentication and update phases unnecessary for CR-PUFs.

A second issue of PUFs in DBPs is the timing requirement posed on the PUF. We propose to exploit the inherent analogous properties of PUFs for more precise time measurements, given the physically minimal runtime of a PUF.

In future work, we want to explore implementation details of our approach by integrating it into an existing DBP and building a proof-of-concept to show the feasibility of the principle and use it to assess performance and scalability. An additional goal is to show the possible distance measurement precision of our DBP, and study the improvement of the fuzzy extractor component being off-loaded to the verifier.

Acknowledgment

We want to thank Matthias Hiller and Michael Pehl from Technische Universität München and Stefan Dietzel from Ulm University for their valuable input on our approach.

References

[1] A. Francillon, B. Danev, and S. Čapkun, "Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars," in *Network and Distributed System Security Symposium (NDSS)*, 2011.

[2] A. Ranganathan, N. Tippenhauer, B. Škorić, D. Singelée, and S. Čapkun, "Design and Implementation of a Terrorist Fraud Resilient Distance Bounding System," in *Computer Security – ESORICS 2012*, Springer, 2012.

[3] A. Yang, Y. Zhuang, and D. S. Wong, "An Efficient Single-Slow-Phase Mutually Authenticated RFID Distance Bounding Protocol with Tag Privacy," in *Information and Communications Security*, Springer, 2012.

[4] C. Kim, G. Avoine, F. Koeune, F.-X. Standaert, and O. Pereira, "The Swiss-Knife RFID Distance Bounding Protocol," in *Information Security and Cryptology – ICISC 2008*, Springer, 2009.

[5] M. Fischlin and C. Onete, "Subtle Kinks in Distance-Bounding: An Analysis of Prominent Protocols," in *Proceedings of the sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, ACM, 2013.

[6] S. Brands and D. Chaum, "Distance-Bounding Protocols," in *Advances in Cryptology – EUROCRYPT '93*, Springer, 1994.

[7] K. B. Rasmussen and S. Capkun, "Realization of RF Distance Bounding," in *USENIX Security Symposium*, 2010.

[8] R. Miesen, R. Ebelt, F. Kirsch, T. Schäfer, G. Li, H. Wang, and M. Vossiek, "Where is the Tag?" *IEEE Microwave Magazine*, vol. 12, no. 7, pp. 49–63, 2011.

[9] J. Clulow, G. P. Hancke, M. G. Kuhn, and T. Moore, "So Near and Yet So Far: Distance-Bounding Attacks in Wireless Networks," in *Security and Privacy in Ad-Hoc and Sensor Networks*, Springer, 2006.

[10] J. Hermans, R. Peeters, and C. Onete, "Efficient, Secure, Private Distance Bounding without Key Updates," in *Proceedings of the sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, ACM, 2013.

[11] G. Hancke and M. Kuhn, "An RFID Distance Bounding Protocol," in *First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, 2005.

[12] J. Reid, J. M. G. Nieto, T. Tang, and B. Senadji, "Detecting Relay Attacks with Timing-based Protocols," in *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security*, ACM, 2007.

[13] R. Plaga and F. Koob, "A Formal Definition and a New Security Mechanism of Physical Unclonable Functions," in *Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*, Springer, 2012.

[14] F. Armknecht, R. Maes, A. Sadeghi, O.-X. Standaert, and C. Wachsmann, "A Formalization of the Security Features of Physical Functions," in *IEEE Symposium on Security and Privacy (SP)*, 2011.

[15] U. Rührmair, J. Sölter, and F. Sehnke, "On the Foundations of Physical Unclonable Functions," IACR Cryptology ePrint Archive, Tech. Rep., 2009.

[16] R. Canetti, O. Goldreich, and S. Halevi, "The Random Oracle Methodology, Revisited," *J. ACM*, vol. 51, no. 4, pp. 557–594, Jul. 2004.

[17] L. Bolotnyy and G. Robins, "Physically Unclonable Function-based Security and Privacy in RFID Systems," in *IEEE International Conference on Pervasive Computing and Communications*, 2007.

[18] M. Bellare and P. Rogaway, "Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols," in *Proceedings of the 1st ACM Conference on Computer and Communications Security*, ACM, 1993.

[19] G. E. Suh and S. Devadas, "Physical Unclonable Functions for Device Authentication and Secret Key Generation," in *Proceedings of the 44th annual Design Automation Conference*, ACM, 2007.

[20] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Silicon Physical Random Functions," in *Proceedings of the 9th ACM conference on Computer and communications security*, ACM, 2002.

[21] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling Attacks on Physical Unclonable Functions," in *Proceedings of the 17th ACM Conference on Computer and Communications Security*, ACM, 2010.

[22] U. Rührmair, J. Solter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas, "PUF Modeling Attacks on Simulated and Silicon Data," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1876–1891, Nov. 2013.

[23] C. Helfmeier, C. Boit, D. Nedospasov, and J.-P. Seifert, "Cloning Physically Unclonable Functions," in *2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2013.

[24] L. Kulseng, Z. Yu, Y. Wei, and Y. Guan, "Lightweight Secure Search Protocols for Low-cost RFID Systems," in *29th IEEE International Conference on Distributed Computing Systems. ICDCS*, 2009.

[25] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, and U. Rührmair, "The Bistable Ring PUF: A New Architecture for Strong Physical Unclonable Functions," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2011.

[26] U. Rührmair, "SIMPL Systems as a Keyless Cryptographic and Security Primitive," in *Cryptography and Security: From Theory to Applications*, Springer, 2012.

[27] J. Rajendran, R. Karri, J. B. Wendt, M. Potkonjak, N. McDonald, G. S. Rose, and B. Wysocki, "Nanoelectronic Solutions for Hardware Security," IACR Cryptology ePrint Archive, Tech. Rep., 2012.

[28] Q. Chen, G. Csaba, X. Ju, S. B. Natarajan, P. Lugli, M. Stutzmann, U. Schlichtmann, and U. Rührmair, "Analog Circuits for Physical Cryptography," in *IEEE International Symposium on Integrated Circuits (ISIC)*, 2009.

[29] S. Kardaş, M. S. Kiraz, M. A. Bingöl, and H. Demirci, "A Novel RFID Distance Bounding Protocol Based on Physically Unclonable Functions," in *RFID. Security and Privacy*, Springer, 2012.

[30] U. Dürholz, M. Fischlin, M. Kasper, and C. Onete, "A Formal Approach to Distance-bounding RFID Protocols," in *Proceedings of the 14th International Conference on Information Security*, Xi'an, China: Springer, 2011.

[31] R.-I. Paise and S. Vaudenay, "Mutual Authentication in RFID: Security and Privacy," in *Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security*, ACM, 2008.