SIMPL Systems, or: Can we design cryptographic hardware without secret key information?

Ulrich Rührmair

Computer Science Department
Technische Universität München
Boltzmannstr. 3
85748 Garching
ruehrmai@in.tum.de
http://www.pcp.in.tum.de

Abstract. This paper discusses a new cryptographic primitive termed *SIMPL system*. Roughly speaking, a SIMPL system is a special type of Physical Unclonable Function (PUF) which possesses a binary description that allows its (slow) public simulation and prediction. Besides this public key like functionality, SIMPL systems have another advantage: No secret information is, or needs to be, contained in SIMPL systems in order to enable cryptographic protocols — neither in the form of a standard binary key, nor as secret information hidden in random, analog features, as it is the case for PUFs. The cryptographic security of SIMPLs instead rests on (i) a physical assumption on their unclonability, and (ii) a computational assumption regarding the complexity of simulating their output. This novel property makes SIMPL systems potentially immune against many known hardware and software attacks, including malware, side channel, invasive, or modeling attacks.

1 Introduction

Background and Motivation. Electronic communication and security devices are pervasive in our life. Just to name two examples, around five billion mobile phones are currently in use worldwide [1] [2], and the world market of smart cards has an estimated volume of over three billion pieces per year [3] [4]. Their widespread use makes such devices both a well-accessible and a worthwhile target for adversaries. Many security attacks are thereby not directed against the employed cryptographic primitives themselves, some of which have proven attack-resilient over surprisingly long time spans. Rather, they often apply physical or software attacks in order to extract the employed secret keys. Such key-extracting strategies are not just a theoretical concern, but have been demonstrated several times in widespread, commercial systems [5] [6] [7]. This drives the search for new mechanisms that protect — or better still: avoid! — secret keys in vulnerable hardware.

Physical Unclonable Functions (PUFs). The security primitive of a Physical Unclonable Function (PUF) [8] [9] [10] [11] was introduced, at least in part, in order to address some of the above problems. A PUF is a (partly) disordered physical system S that can

be challenged with so-called external stimuli or challenges C_i, upon which it reacts with corresponding responses termed R_{C_i}. Contrary to standard digital systems, a PUF's responses shall depend on the nanoscale structural disorder present in the PUF. This disorder cannot be cloned or reproduced exactly, not even by its original manufacturer, and is unique to each PUF. Assuming the stability of the PUFs responses, any PUF S hence implements an individual function F_S that maps challenges C_i to responses R_{C_i} of the PUF. Due to its complex and disordered structure, a PUF can avoid some of the shortcomings associated with digital keys. For example, it is usually harder to read out, predict, or derive its responses than to obtain the values of digital keys stored in non-volatile memory. This fact has been exploited for various PUF-based security protocols [8] [9] [15] [22].

One prominent example are PUF-based identification schemes [8] [9] [10]. They are usually run between a central authority (CA) and a hardware carrying a (unique) PUF S. One assumes that the CA had earlier access to S, and could establish a large, secret list of challenge-response-pairs (CRPs) of S. Whenever the hardware wants to identify itself to the CA at some later point in time, the CA selects some CRPs at random from this list, and sends the challenges contained in these CRPs to the hardware. The hardware applies these challenges to S, and sends the obtained responses to the CA. If these responses match the pre-recorded responses in the CRP-list, the CA believes the identity of the hardware. Note that each CRP can only be used once, whence the CRP-list uses up over time, and needs to be large.

Private Key like Functionality of PUFs. The described protocol has several well-known advantages [8] [9]. However, one potential downside is that it presumes a previously shared piece of secret numerical information (i.e., the CRP-list). This information needs to be established in a secure set-up phase between the CA and the hardware, and must constantly be kept secret. In this particular structural aspect, PUFs are resemblant of classical private key systems.

Secret Information in PUFs. Another noteworthy point is that PUFs in general do not obviate the presence of secret information within cryptographic hardware. The secret information is no longer stored in digital form in two-level systems, such as digital secret keys stored in non-volatile memory cells. But still there is some sort of secret information present in most PUFs, whose disclosure breaks the security of the system. Let us name two examples to illustrate our point: In the case of SRAM PUFs the information that needs to be kept secret is the state of the SRAM cells after power up, or the tiny manufacturing variations of the SRAM cells that determine their state after power up [25]. Once this information is known to an adversary, he can numerically derive the same key as the cryptographic hardware embedding the SRAM PUF. In the case of Arbiter PUFs, the secret information are the internal runtime delays in the circuit stages [11]. If this information is known, the adversary can numerically simulate the behavior of the PUF output by an additive, linear model, again breaking its security [26].

In other words, the architectures of most current PUFs "hide" or "obfuscate" secret, security-relevant information very well in analog characteristics of integrated circuits. But at the same time, they do not avoid the need for secret information in hardware systems in principle; they just store it in a different form.

Our Contributions. This paper introduces a novel security primitive called a *SIMPL system*, whereby the acronym SIMPL stands for "SIMulation Possible, but Laborious". These systems have two interesting conceptual advantages: First, they are a PUF-like security tool, but possess some type of public key functionality. This improves their practical applicability. Second, they obviate the need for secret information in cryptographic systems, trading it for two other assumptions: (i) their physical unclonability, and (ii) the assumed computational overhead of numerically simulating their output (in comparison with their faster real-time behavior). We show that SIMPLs can realize basic communication protocols such as identification and message authentication, and briefly describe the application of these protocols in some concrete settings. We also discuss implementations of SIMPL systems, thereby surveying existing approaches, and propose a new optical implementation strategy. Proof-of-concept data on this optical implementation, which arose from other, recent research activities in our group [40], is presented in the appendix.

Related Work. The current paper is an extended version of [16]. Since the publication of [16], several papers of our group have dealt with the implementation of SIMPLs by integrated circuits [17] [18] [19] [20]. We emphasize that around the same time as [16], a comparable concept has been described independently in [21] under the name of a Public PUF (PPUF).

While stressing that both pieces of work are very interesting, let us briefly address a few differences between [21] and our studies. One difference is that we focus on SIMPL systems/Public PUFs for which the *relative* speed difference between the real hardware and the simulation is comparably low, for example only a small constant factor. Such systems seem to have milder complexity requirements and less stability issues. We argue that by applying feedback loops, not the relative, but the absolute time difference between such systems and any emulation can still be amplified to a sufficient absolute value. Once this absolute value is large enough, it enables secure identification and message authentication protocols, and could compensate network or other delays. Another reason for concentrating on systems with small speed gaps lies in the fact that the verification step in identification and message authentication must be carried out relatively efficiently (see Protocols 2 and 3).

Second, we center upon applications where the main advantage of SIMPL systems — that they can build security systems without secret key information — is most relevant. Two typical examples are the named identification and message authentication schemes. Should a shared secret key between two parties be required in a SIMPL-based communication infrastructure (for example in order to achieve confidentiality), SIMPL-based message authentication can be used together with the Diffie-Hellman protocol to exchange a session key. But this key ideally will not be stored permanently in the system.

To the contrary, [21] discuss a PPUF-scenario where a one-time, permanent secret key is exchanged in a computationally relatively intensive scheme. This scheme appears too time consuming for multiple session key exchange. Their setting hence puts key exchange on different security assumptions than classical protocols (like Diffie-Hellman), which is a strong achievement on its own. But they do not attempt to generally avoid

the long-term presence of secret information in cryptographic hardware, as we aspire with SIMPL systems.

Finally, a very interesting and recommendable, but later source is [24], where time-bounded authentication for FPGAs is discussed.

Organization of this Paper. The rest of this manuscript is organized as follows: In Section 2, we give a semi-formal specification of SIMPL systems, and discuss their properties. Section 3 provides two formal SIMPL-based protocols for entity identification and message authentication. In Section 4 we discuss implementation candidates for SIMPL systems, and conclude the paper in Section 5.

2 SIMPL Systems and their Properties

2.1 Informal Description of SIMPL Systems

We start by informally listing the properties of a SIMPL system. A physical system S is called a *SIMPL system* (or just a *SIMPL*) if the following holds:

1. S is a (partly) disordered physical system. It can be stimulated with challenges C_i, upon which it reacts with corresponding responses R_{C_i}. The responses are a function of the specific disorder present in S, and of the applied challenge C_i. The responses are assumed to be sufficiently stable to regard the behavior of S as a function F_S that maps challenges C_i to responses R_{C_i}.
2. Given a challenge C_i, it is possible to numerically simulate the corresponding response R_{C_i} of S with high accuracy. The simulation is carried out via an individual, public description $D(S)$ of S, and a public simulation algorithm Sim.
3. Any feasible algorithm, or any physical emulation, that predicts the responses of S correctly (i.e., which computes F_S), is noticeably slower than the real-time behavior of S.
4. It is difficult to physically clone S, i.e. to produce a second system S' which generates the same responses on almost all possible challenges with comparable speed. This must hold even if the internal characteristics and disorder of S, the description $D(S)$, and many CRPs of S are known.

Put in one sentence, the holder of a secure SIMPL system S is able to evaluate a publicly known, publicly computable individual function F_S *faster* than anyone else.

2.2 Semi-Formal Specification of SIMPL Systems

The above properties can also be coined into a semi-formal specification of SIMPL systems. The style of the specification follows the specifications and definitions that have been presented in [22]. It specifies the security of SIMPL systems as a “game” with the adversary, thereby introducing a relatively precise adversarial model.

Specification 1 (($t_{max}, c, t_C, t_{Ph}, q, \epsilon$)-SIMPL SYSTEMS.)**.** *Let S be a physical system mapping challenges C_i to responses R_{C_i}, with $\mathbf{C}$ denoting the finite set of all possible challenges. Let $c > 1$ be a constant, and let furthermore t_{max} be the maximum time (over all challenges $C_i \in \mathbf{C}$) which it takes until the system S has generated the response R_{C_i} to the challenge C_i. S is called a $(t_{max}, c, t_C, t_{Ph}, q, \epsilon)$-*SIMPL SYSTEM *if there is a string $D(S)$, called the description of S, and a computer algorithm* Sim *such that the following conditions are met:*

1. *For all challenges $C_i \in \mathbf{C}$, the algorithm* Sim *on input $\big(C_i,\ D(S)\big)$ outputs R_{C_i} in feasible time.*
2. *Any cryptographic adversary Eve will succeed in the following* **security experiment** *with a probability of at most ϵ:*
   (a) Eve is given the numerical description $D(S)$ and the code of the algorithm Sim *for a time period of length t_C.*
   (b) Within the above time period t_C, Eve can q times adaptively query an oracle O for arbitrary responses R_{C_i} of S .
   (c) Within the above time period t_C, Eve is furthermore given physical access to the system S at adaptively chosen time points, and for time periods of adaptively chosen lengths. The only restriction is that her access times must add up to a total of at most t_{Ph}.
   (d) After the time period t_C has expired, Eve is presented with a challenge C_{i_0} that was chosen uniformly at random from the set $\mathbf{C}$, and is asked to output a value V_{Eve}.

   We say that Eve succeeds in the described experiment if the following conditions are met:
   (i) $V_{Eve} = R_{C_{i_0}}$.
   (ii) The time that Eve needed to output V_{Eve} after she was presented with C_{i_0} was at most $c \cdot t_{max}$.

   Please note that the said probability of ϵ is taken over the uniformly random choice of $C_{i_0} \in \mathbf{C}$, and the random choices or actions that Eve might take in steps 2a, 2c and 2d.

Discussion. Let us briefly discuss the security model underlying Specification 1. In practical applications of SIMPL systems, Eve can gather information about S in three ways: (i) She analyzes the algorithm Sim and the description $D(S)$, which are both public. (ii) She collects as many challenge-response-pairs (C_i, R_{C_i}) of S as possible from external sources, for example protocol eavesdropping. (iii) Eve *physically* measures the system S. She may determine CRPs by such measurements, but also other, more general characteristics of the system.

These three types of attacks must be covered in our security model, and they are: Possibility (i) is covered in item *2a* of Spec. 1, (ii) is reflected in item *2b*, and (iii) is implicit in item *2c*. Since the physical access time and the time in which Eve can prepare her attack by previous computations differ strongly in most application scenarios, it makes sense to distinguish between t_{Ph} and t_C in Spec. 1.

We also chose the value c, which describes the time gap between Eve and the SIMPL system, to be a flexible system parameter. This keeps the definition general and allows

its application to different types of SIMPLs. In many practical applications, even small values (e.g. around 2) may suffice for c. See also the discussion in Section 2.3, paragraphs on *constant vs. super-polynomial time gap* and *feedback loops*.

2.3 Properties of SIMPL Systems

Let us discuss a few properties of SIMPL systems implied by Specification 1.

Immunity against ϵ-fraction Read-out and Simulation. Spec. 1 implies that for any SIMPL system it must be impossible to measure the values R_{C_i} for more than an ϵ-fraction of all parameters $C_i \in \mathbf{C}$ within time t_{Ph}. Otherwise, Eve could create a lookup table for an ϵ-fraction of all possible values R_{C_i} during step 2c. This could enable her to succeed in the described experiment with probability greater than ϵ. Therefore, for any SIMPL system the set of possible measurement parameters $\mathbf{C}$ must be very large.

For the same reasons, it must be impossible for Eve to determine more than an ϵ-fraction of all CRPs within time t_C by exhaustive simulation on the basis of Sim and $D(S)$. This again implies that $\mathbf{C}$ must be very large (for example exponential in some system parameter), and/or that the simulation must be time consuming.

Immunity against Cloning. Spec. 1 also implies that previous physical access for time t_{Ph} and computations of time t_C do not allow Eve to build a *physical clone* S' of the system S, for whose responses R'_{C_i} it holds that

$$R_{C_i} = R'_{C_i} \quad \text{for more than an } \epsilon\text{-fraction of all } C_i \in \mathbf{C},$$

and for which the evaluation of the R'_{C_i} works within time $c \cdot t_{max}$. Spec. 1 both rules out the possibility to build an exact physical reproduction of S, or the feasibility to fabricate a *functional* clone, i.e., a physical system of a possibly very different structure or lengthscale than S, which still generates its response R'_{C_i} within time $c \cdot t_{max}$.

Constant vs. Super-polynomial Time Gap. Spec. 1 stipulates that the time gap between Eve and the real SIMPL system S must be at least a constant factor $c > 1$. This seems surprising: Being used to the formalism of complexity-based classical cryptography, one might expect the stipulation of an exponential gap. But it is unclear whether SIMPLs with an exponential time margin between Eve and the SIMPL exist at all. The only known, realistic computational systems which might outperform Turing architectures by a super-polynomial factor are quantum computers [35]. But standard quantum computers possess no immunity against physical cloning, since they could be mass-fabricated with the same functionality. They are hence unsuited as SIMPL systems. A further setback in the search for SIMPLs with an exponential security margin is that it has been frequently hypothesized within the computational complexity community that there are no realistic hardware systems that solve NP-complete problems efficiently in practice, i.e. by using polynomial resources. Two recent sources in this context are [33] [34].

Still, meaningful applications for SIMPL systems may not require exponential speed gaps. In the appliances we suggest in this paper (namely identification and on-the-fly

message authentication), a constant, detectable time difference suffices. An exponential time gap between the SIMPL system and any simulation machine may even be undesirable there, since it could lead to time consuming verification steps in the Protocols 2 and 3.

Feedback Loops. In order to enable large absolute time margins, the absolute (but not the relative!) time difference between the original SIMPL system and any fraudster can be amplified via feedback loops. In a nutshell, such feedback-loops can be set up as follows: Presented with a challenge C_1, the SIMPL systems successively determines a sequence of k challenge-responses-pairs $(C_1, R_{C_1}), (C_2, R_{C_2}), \ldots, (C_k, R_{C_k})$, where later challenges C_n are determined by earlier results R_{C_m}, with $k \geq n > m \geq 1$. The tuple (C_1, R_{C_k}) can then be regarded as the overall challenge-response pair determined by the SIMPL. The application of such feed-back loops can help us to compensate network and transmission delays.

Let us make a concrete example in order to illustrate our point. Suppose that we possess a SIMPL system S which produces its responses in t_{max} of 10 nanoseconds (ns), and which possesses a speed advantage of $c = 2$ over all simulations. That means that any adversary cannot produce the response to a randomly chosen challenge within 20 ns. This tiny difference would not be detectable in many practical settings, for example in large networks with natural delays. Nevertheless, the application of repeated feedback loops can amplify not the relative, but the absolute time margin, such as to 1 millisecond (ms) vs. 2 ms, or also 1 sec vs. 2 sec.

SIMPLs with Multi-bit Output. In some applications, it is found convenient if a SIMPL system produces not just one bit as response, but a multi-bit output. Some implementations of SIMPLs have this property naturally (for example the optical implementation of section 4.3). Otherwise, feedback loops can allow us to create multi-bit outputs from SIMPL systems with 1-bit outputs: One simply considers a concatenation (or some other function, for example a hash function) of the last n responses $R_{C_{k-n+1}}, \ldots, R_{C_k}$ in the feedback loop. This concatenation (or function) can be taken as the overall output of the SIMPL.

Another option to create "large" SIMPL systems with k-bit outputs from "small" SIMPL systems with 1-bit outputs is to employ k such SIMPL systems in parallel, and to directly concatenate their responses to produce a k-bit overall output. This method has been suggested already in the context of PUFs in [13].

Error Correction. Please note that in the Spec. 1, in the above discussion, and also in the upcoming protocols in Section 3, we assume that the responses of the SIMPL system are stable. In practice, error correction and helper data must, and can, be applied to achieve this goal; see, for example, [9] [37] [38] [39].

3 Protocols and Applications

We will now describe two exemplary protocols that can be realized by SIMPL systems, and discuss some application scenarios.

3.1 Identification of Entities

We assume that Alice holds an individual $(t_{max}, c, t_C, t_{Ph}, q, \epsilon)$-SIMPL system S, and has made the corresponding data $D(S)$, Sim, the value $c \cdot t_{max}$, and a description of $\mathbf{C}$ public. Now, she can prove her identity to an arbitrary second party Bob, who knows $D(S)$, Sim, $c \cdot t_{max}$ and $\mathbf{C}$, as follows (with k being the security parameter of the protocol):

Protocol 2: IDENTIFICATION OF ENTITIES BY SIMPL SYSTEMS

1. Bob chooses k challenges $C_1, \ldots, C_k$ uniformly at random from $\mathbf{C}$.
2. **For** $i = 1, \ldots, k$ **do**:
   (a) Bob sends the value C_i to Alice.
   (b) Alice determines the corresponding response R_{C_i} by an experiment on her SIMPL system S, and sends this value to Bob.
   (c) Bob receives an answer from Alice, which we denote by V_i. If Alice's answer did not arrive within time $c \cdot t_{max}$, then Bob sets $V_i = \perp$ and continues the for-loop.
3. Bob computes the value $R_{C_i}^{Sim} = \mathsf{Sim}(C_i, D(S))$ for all $i = 1, \ldots, k$, and verifies if $R_{C_i}^{Sim} = V_i \neq \perp$. If this is the case, Bob believes Alice's identity, otherwise not.

Discussion. In a nutshell, the security of the protocol follows from the fact that an adversary is unable to determine the values R_{C_i} for randomly chosen C_i comparably quickly as Alice, provided that: (i) The lifetime of the system S (and the period since $D(S)$ was made public) does not exceed t_C, and (ii) the adversary's accumulated physical access times do not exceed t_{Ph} (see Spec. 1). In that case, the adversary's probability to succeed in the protocol without possessing S decrease exponential in k.

Bob can improve his computational efficiency by verifying the correctness of the responses R_{C_i} only for a randomly chosen subset of all responses. If necessary, possible network and transmission delays can be compensated for by amplifying the absolute time gap between Eve and S through feedback loops (see Section 2.3).

If the SIMPL system has multi-bit output, then a value of $k = 1$, i.e. a protocol with one round, may suffice. In these cases, the parameter ϵ of the multi-output SIMPL system will in itself be exponentially small in some system parameter (for example in the size of the sensor array in the optical SIMPLs discussed in Section 4).

3.2 Authentication of Messages

Alice can also employ an individual $(t_{max}, c, t_C, t_{Ph}, q, \epsilon)$-SIMPL system S in her possession to authenticate messages to Bob. Again, we suppose that the values $D(S)$, Sim, $c \cdot t_{max}$, and a description of $\mathbf{C}$ are public.

Protocol 3: AUTHENTICATION OF A MESSAGE N BY SIMPL SYSTEMS

1. Alice sends the message N, which shall be authenticated, to Bob.

2. Bob chooses $k \cdot l$ challenges $C_1^1, \ldots, C_k^1, C_1^2, \ldots, C_k^2, \ldots, C_1^l, \ldots, C_k^l$ uniformly at random from $\mathbf{C}$.
3. **For** $i = 1, \ldots, l$ **do**:
   (a) Bob sends the values $C_1^i, \ldots, C_k^i$ to Alice.
   (b) Alice determines the corresponding responses $R_{C_1^i}, \ldots, R_{C_k^i}$ by experiments on her SIMPL system S.
   (c) Alice derives a MAC-key K_i from $R_{C_1^i}, \ldots, R_{C_k^i}$ by a publicly known procedure, for example by applying a publicly known hash function to these values. She sends $MAC_{K_i}(N)$ to Bob.
   (d) Let us denote the answer Bob receives from Alice by V_i. If V_i did not arrive in time $c \cdot t_{max} + t_{MAC}$, where t_{MAC} is the time to derive K_i and compute $MAC_{K_i}(N)$, then Bob sets $V_i = \perp$ and continues the for-loop.
4. For $i = 1, \ldots, k$ and $j = 1, \ldots, l$, Bob computes the values $R_{C_i^j}^{Sim} = \mathsf{Sim}(C_i^j, D(S))$ by simulation via Sim. He derives the keys $K_1^{Sim} \ldots, K_k^{Sim}$ by application of the same procedure (e.g. the same publicly known hash function) as Alice in step 3c.
5. For all $i = 1, \ldots, k$, Bob checks if it holds that $MAC_{K_i^{Sim}}(N) = V_i \neq \perp$. If this is the case, he regards the message N as properly authenticated, otherwise not.

Discussion. In a nutshell, the security of the protocol follows from the fact that an adversary cannot determine the responses $R_{C_i^j}$ and the MAC-Keys $K_1, \ldots, K_l$ as quickly as Alice. As earlier, verification of a randomly chosen subset of all MACs can improve Bob's computational efficiency in step 5. Depending on the exact circumstances, a few erroneous V_i may be tolerated in step 5, too.

We assume without loss of generality that the MAC can be computed quickly (including the derivation of the MAC keys $K_1, \ldots, K_l$), i.e., within time t_{MAC}, and that t_{MAC} is small compared to t_{max}. Again, this condition could be realized by amplification through feedback loops if necessary (see Section 2.3). Furthermore, it is known that MACs can be implemented very efficiently [27]. If information-theoretically secure hash functions and MACs are used, the security of the protocol will not depend on any assumptions other than the security of the SIMPL system.

If the SIMPL system has a multi-bit output, then values of $k = 1$, i.e., sending just one challenge in each round, or of $l = 1$, i.e., employing just one round of communication, may suffice. Such a multi-bit output can arise either naturally, for example through the choice of the SIMPL system itself (as noted earlier, the optical SIMPL system presented in Section 4.3 has this property). Or it can be enforced by feedback loops, or by using several independent SIMPL systems in parallel (see Section 2.3, page 7). In fact, such measures even are strictly necessary to uphold the protocol's security if the constant c has got a very low value.

3.3 Application Scenarios

Secure Communication Infrastructures. Within the given space restrictions, we will now discuss the application of SIMPL systems to secure communication in networks, illustrating their potential in such a setting. Consider a situation where k parties $P_1, \ldots, P_k$ and a trusted authority TA participate in a communication network. Assume that each

party P_i carries its own SIMPL S_i in its hardware, and that a certificate C_i has been issued for each party by the TA. The certificate includes the identity and the rights of Party P_i, and has the form

$$C_i = \big(Id_i, Rights_i, D(S_i), Sig_{TA}(Id_i, Rights_i, D(S_i))\big).$$

Under these provisions, the parties can mutually identify themselves by Protocol 2, they can establish authenticated channels with each other by Protocol 3, and they can exchange session keys via the Diffie-Hellman protocol [32] over these authenticated channels. The whole architecture works without permanent secret keys, or without any other secret information that is stored permanently in the hardware of the parties $P_1, \ldots, P_k$.

It also seems well applicable to cloud computing: All personal data could be stored centrally. Session keys could be exchanged by the Diffie-Hellman protocol over channels authenticated by the SIMPL systems. These keys can be used to download the personal data in encrypted form from the central storage. The keys can be new in each session, no permanent secret keys in the mobile hardware are be necessary.

The above approaches can further be combined with tamper-sensitive SIMPL systems. These SIMPLs may cover hardware which has a functionality $Func_i$ as long as it is non-manipulated. Each certificate C_i could then also include the functionality of the hardware, i.e., it could be of the form

$$C_i = \big(Id_i, Rights_i, Func_i, D(S_i), Sig_{TA}(Id_i, Rights_i, Func_i, D(S_i))\big).$$

By running the identification protocol (Prot. 2), party P_i can prove to party P_j that the SIMPL system S_i is non-tampered, and that the hardware hence has the claimed functionality $Func_i$. Please note that the optical SIMPL systems we propose in this paper is naturally tamper sensitive; the tamper sensitivity of such optical scattering structures has already been shown in detail in [8].

Two other Applications. Let us, in all brevity, point to two other applications of SIMPL systems. They are described in more detail in [16].

A first application is the generation of unforgeable labels for products or security tokens. SIMPL systems can create labels which do not contain any secret information, which can be verified offline, and which only require remote, digital communication between the label and a testing device. These properties are not met by other known labeling techniques: RFID-tags with secret keys obviously contain secret information; PUF-based labels contain secret information in the case of Weak PUFs, and require an online database in the case of Strong PUFs [8]; and current Certificates of Authenticity (COAs) [28] [30] require analog near-field measurements in the verification step.

Another application area of SIMPLs lies in the context of the digital rights management problem. SIMPLs can create unclonable representations of digital content [16]. Similar to the unforgeable labels, these unclonable representations of digitial content do not contain any secret information. They can be verified for their validity offline and by mere digital communication between a tester and the device carrying the unclonable representation. Again, in combination these features are not met by any comparable technique known to the author. In [29] [30] [31], for example, the random features of the data carrier must be determined in the near-field by analog measurements.

4 Implementation of SIMPL Systems

Let us now turn to the practical implementation of SIMPL systems. We will give an overview of existing ideas and challenges, and propose one new, optical concept.

4.1 Challenges

It turns out that there are some strong challenges in the realization of SIMPL systems. The three non-trivial requirements that need to be balanced are complexity, stability, and simulatability: On the one hand, the output of a SIMPL system must be sufficiently complex to require a long computation/simulation time. On the other hand, it must be simple enough to allow simulation at all, and to enable the determination of $D(S)$ by measurement or numeric analysis techniques. A final requirement is that the simulation can be carried out *relatively* efficiently by everyone (this is necessary to complete the verification steps in the identification and message authentication protocols quickly); while, at the same time, even a very well equipped attacker, who can potentially attempt to parallelize the simulation on many powerful machines, cannot simulate as fast as the real-time behavior of the SIMPL system. In the sequel, we will discuss a few implementations that try to meet these seemingly conflicting requirements.

4.2 Electrical SIMPL Systems

Since the first publication of [16], a sequence of papers of our group has dealt with the implementation of SIMPL systems by electrical, integrated circuits [17] [18] [19] [20]. We tried to exploit two known speed bottlenecks of modern CPUs: Their problems in dealing simultaneously with very large amounts of data, and the complexity of simulating analog, parallel phenomena. Let us briefly summarize these approaches, quoting from said papers.

"Skew" SRAM Memories. A first suggestion made in [17] [18] [19] [20] is to employ large arrays of SRAM cells with a special architecture named "skew design". In this design, the read- and write behavior of the cells is dependent on the applied operational voltage. The simulation of a skew SRAM memory in a feedback loop of a very large number of successive read- and write events then seems somewhat laborious to simulate on a standard architecture. The hypothesis put forward in [17] [18] [19] [20] is that this creates a small, constant simulation overhead. Two essential assumptions in this concept are: (i) No parallelization is possible, since the successive read- and write events in the feedback loop are made dependent on the previous read results. And (ii), since no parallelization is possible, the limiting factor for an adversary is his clock frequency, which is quite strongly limited by current technology.

As described in the listed references, the idea shows strong promise to succeed against any adversaries with a limited financial budget, and in particular against any FPGA-based attacks. Future work will need to show how large the exact simulation margin is, and whether it is indeed sufficient to defeat an adversary with large resources, who is capable of fabricating ASICs. Due to its relatively easy realizability and good security level, the idea could have a strong potential for the consumer market.

Two-dimensional Analog Computing Arrays. A second suggestion of [17] [18] [19] [20] consists of using analog, two-dimensional computing arrays. The authors suggest the use of so-called cellular non-linear networks (CNNs) which are designed to imitate non-linear optical systems. Due to their analog and inherently parallel nature (many cells exchange information at the same time), it is suggested that CNNs are time consuming to simulate on a digital, sequential architecture.

This idea has its assets on the security side: Since it is based on manufacturing mismatches in CNN fabrication that currently seem unavoidable, it shows promise of defeating even attackers with very strong financial resources, and of being manufacturer resistant in the sense of [23]. It requires the use of analog circuits, though, which might potentially be unsuited for low-cost applications.

Other Approaches. Independently, the work of other groups has lead to different structures that could be used as SIMPLs. The implementation of PPUFs presented in [21] could potentially be downscaled to become a SIMPL system, even though it would have to be carefully investigated how resilient such small-scale instances are against parallelization attacks. Another very interesting, FPGA-based candidate for SIMPLs is implicit in the work of [24].

4.3 Integrated Optical SIMPLs

Also optical structures can be used as SIMPL systems. The rationale behind employing optics is as follows: First, optical systems can potentially achieve faster component interaction than electronic systems; this promises to create the desired speed advantage over any electronic simulator. The phenomenon of optical interference has no electronic analog at room temperature [41], and can create a computational overhead for electronic simulators. Second, the material degradation of optical systems is low, and their temperature stability is known to be high [41] [42]. Even very complex and randomly structured optical systems, whose internal complexity creates the desired speed gaps, can produce outputs that are stable against aging and environmental conditions.

The concrete optical SIMPL we suggest is depicted schematically in Figure 1. It comprises of an immobile laser diode array with k phase-locked diodes $D_1, \ldots, D_k$ [43], which is used to excite a disordered, random scattering medium. The diodes can be switched on and off independently, leading to 2^k challenges C_i. These can be written as $C_i = (b_1, \ldots, b_k)$, where each $b_i \in \{0, 1\}$ indicates whether diode D_i is switched on or off. (Note that the diode array must indeed be phase locked in order to allow interference of the different diode signals.) At the right hand side of the system, an array of l light sensors $S_1, \ldots, S_l$, e.g. photodiodes, measures the resulting light intensities locally. A response R_{C_i} consist of the intensities $I_1, \ldots, I_l$ in the l sensors. Instead of phase-locked diode arrays, also a single laser source with a subsequently placed, inexpensive light modulator (as contained in any commercially available beamer) can be employed.

Under the provision that a *linear* scattering medium is used in such integrated optical SIMPLs, the following analysis holds. Every diode D_i with $b_i = 1$ creates a lightwave, which is scattered in the medium and arrives at the sensor S_j with amplitude

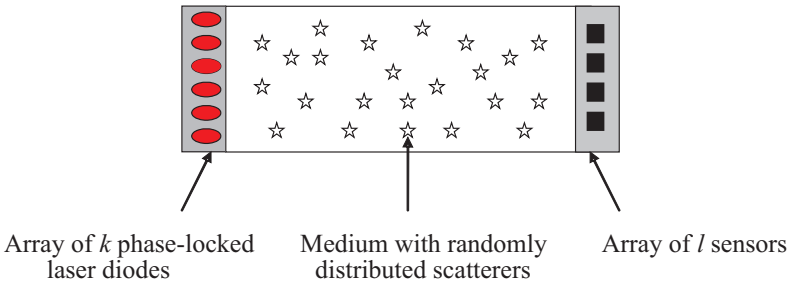

Fig. 1. An integrated optical SIMPL system

E_{ij} and phase shift θ_{ij}. The intensity I_j at the sensor S_j is then given by [42]

$$I_j = \left|E_j\right|^2 = \left|\sum_i b_i \, E_{ij} \, \cos\, \theta_{ij}\right|^2. \tag{1}$$

For the said linear scattering medium, the amplitude E_{ij} and phase shift θ_{ij} are independent of whether the other diodes are switched on or off. One can hence collect many CRPs

$$(C_m, R_{C_m}) = ((b_1, \ldots, b_k), (I_1, \ldots, I_l)),$$

and derive the values E_{ij} and θ_{ij} from knowledge of these many (C_m, R_{C_m}). One suited approach are machine learning techniques, for example a standard machine learning regression.

Once the parameters E_{ij} and θ_{ij} are known, the simulation of a response $R_{C_m} = (I_1, \ldots, I_l)$ to a given challenge $C_m = (b_1, \ldots, b_k)$ can be executed by simple calculation following Eqn. 1. The time margin to the real system will be small, but likely detectable: The real system creates its output and the complex interference in nanoseconds, while the calculation of Eqn. 1 requires around $k \cdot l$ multiplications and $k \cdot l$ additions. Some of these computations can be parallelized, and the values $E_{ij} \cdot \cos\theta_{ij}$ can be precomputed. Still, even for a moderate size of the two-dimensional diode and sensor arrays of around $100 \times 100 = 10^4$ each, the number of additions is on the order of 10^8. This seems to create exactly the constant, notable time gap that we require in SIMPLs.

A first proof-of-concept for this integrated optical approach, which is not optimized in terms of speed, but shows the feasibility of the output simulation/prediction on the basis of real data, is given in the appendix.

4.4 Further Implementation Strategies

Let us discuss a few further implementation strategies for SIMPLs.

Employing PUFs with Reduced Complexity. One generic strategy for the realization of SIMPL systems, which has been suggested already in [16], is the following: Employ a

PUF or a PUF-like structure; and reduce its inner complexity until it can be characterized by measurements and simulated, or until it can successfully be machine learned. If the level of complexity is still sufficient, then this simulation will be more time consuming than the real-time behavior of the system. In fact, the suggestions of the previous subsections used this strategy already, since both CNNs and integrated optical structures have already been suggested as PUFs in earlier work [36] [12]. But also any other PUFs could be used in this strategy, for example Pappu's original optical PUF with a reduced number of scatterers [8], as suggested in [16].

Simulation vs. Verification. Another interesting idea is to exploit the well-known asymmetry between actively computing a solution for a certain problem and verifying the correctness of a proposed solution (as also implicit in the infamous P vs. NP question) [16]. Exploiting this asymmetry could lead to protocols of the following kind: A SIMPL system provides the verifier in an identification/authentication protocols with some extra information that allows the verifier to *verify* its answers fast. To illustrate our point, imagine an analog, two-dimensional, cellular computing array whose behavior is governed by partial differential equations (PDEs), such as the CNN described in section 4.2. Then, verifying the validity of a given final state of such a PDE-driven system (i.e. verifying that this state is indeed a solution of the PDEs driving the system) could be much more time efficient than computing this solution from scratch. Furthermore, the verifier could not only be given external outputs of such a two-dimensional array (e.g. values in boundary cells), but also internal submeasurements (e.g. values in inner cells) that help him to verify the output quickly.

The simulation vs. verification strategy can help to relieve the seeming conflict between the requirement for fast simulation on the side of the verifier (who may not be well equipped on the hardware side) and the necessary time margin to an attacker (who may be very well equipped on the hardware side), which we already addressed in Section 4.1.

5 Summary, Discussion, and Future Work

Summary. This paper introduced a security concept termed "SIMPL Systems". We started by a explaining the basic idea and by giving a semi-formal specification of SIMPL systems. We subsequently discussed some basic properties that follow from this specification. We then presented two protocols that can be realized by SIMPL systems, namely identification and message authentication. These protocols exploit the fact that the holder of a SIMPL system is the only person who can determine the response of the SIMPL to a randomly chosen challenge within a certain time frame. We argued that the can be used to set up special, secure communication infrastructures which obviate the long-term storage of any form of secret keys in hardware. We listed other applications of SIMPL systems, for example as unforgeable labels and in the digital rights management problem.

We next discussed the practical implementation of SIMPL systems. We gave an overview of existing, electrical candidates, and then suggested a new optical implementation based on light scattering. We gave a proof-of-concept for this optical SIMPL by

using data from a first prototype, which had been set-up by our group in a different context [40]. This data shows the general feasibility of predicting such systems, but was not yet optimized in terms of speed. We also presented generic and/or future implementation strategies for SIMPLs, for example the use of PUFs with reduced complexity, or exploiting the asymmetry between actively computing and merely verifying a solution to a given problem (as implicit in the well-known P vs. NP question).

Discussion. Let us conclude this work by a detailed comparative analysis. As said earlier, there are some similarities between classical private/public key cryptoschemes and SIMPL systems: The numeric description $D(S)$ is some analog to a public key, while the physical system S itself constitutes some functional equivalent to a private key. This provides SIMPLs with some public-key like functionality and with the resulting practicality advantages.

At the same time, there is one important difference to classical public-key systems: This new type of "private key" S is no secret numeric information, but a randomly structured, hard-to-clone *physical system*. It has the interesting feature of not containing any form of secret information. Neither in an explicit digital form like a digital key in classical hardware. Nor in a hidden, analog form such as internal PUF parameters (for example the mentioned delay values in the Arbiter PUFs, or the parameters determining SRAM behavior). All internal characteristics of a SIMPL, including its precise internal configuration, can be publicly known without compromising the security of the derived cryptographic protocols.

The security of SIMPL systems is not free of assumptions, though. Instead of presupposing the secrecy of some sort of information, it rests on the following two hypotheses: (i) on the computational assumption that no other, well-controllable, configurable, or even programmable hardware can generate the complex responses of a SIMPL with the same speed, and (ii) on the physical assumption that it is practically infeasible for Eve to exactly clone or rebuild the SIMPL system, even though she knows its internal structure and properties. [1]

It is long accepted that computational assumptions play a standard role in classical cryptography, and they are also a part of the security assumptions for SIMPL systems; but SIMPLs show that one can trade the need for secret information in the hardware against assumptions on the physical unclonability of the SIMPL system. This can surprisingly obviate the familiar requirement that cryptographic hardware must contain secret key information of some sort.

Future Work and Prospects. Future work on SIMPLs will likely concentrate on new protocols beyond identification and message authentication, and on formal security proofs for such protocols. But perhaps the greater challenge lies on the hardware side: Even though there are several promising candidates (see Section 4), the issue of finding a highly secure, practical, and cheap implementation of SIMPL systems appears not to

[1] The reader can verify the plausibility of the latter unclonability property by considering the optical implementation of section 4.3: Even if the positions of all scattering centers and the other irregularities in the plastic matrix were known in full detail, it would still be infeasible to rebuild the whole system with perfect precision.

be fully settled yet. If such an implementation is found, or if the existing implementation candidates are shown to possess all necessary properties, this could change the way we exercise cryptography today.

References

1. http://www.cbsnews.com/stories/2010/02/15/business/main6209772.shtml
2. http://www.bbc.co.uk/news/10569081
3. http://www.eurosmart.com/images/doc/Eurosmart-in-the-press/2006/cardtechnologytoday_dec2006.pdf
4. http://www.gsaietsemiconductorforum.com/2010/delegate/documents/GASSELGSALondon20100518presented.pdf. Slide 23.
5. T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, M. T. Manzuri Shalmani: *On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme.* CRYPTO 2008.
6. T. Kasper, M. Silbermann, C. Paar: *All You Can Eat or Breaking a Real-World Contactless Payment System*. Financial Cryptography and Data Security, January 25-28, 2010.
7. R. J. Anderson: *Security Engineering: A Guide to Building Dependable Distributed Systems, 2nd Edition.* ISBN: 978-0-470-06852-6, Wiley, 2008.
8. R. Pappu, B. Recht, J. Taylor, N. Gershenfeld, *Physical One-Way Functions*, Science, vol. 297, pp. 2026-2030, 20 September 2002.
9. R. Pappu, *Physical One-Way Functions*, PhD Thesis, MIT.
10. B. Gassend, D. E. Clarke, M. van Dijk, S. Devadas: *Silicon physical random functions.* ACM Conference on Computer and Communications Security 2002, 148-160.
11. B. Gassend, D. Lim, D. Clarke, M. v. Dijk, S. Devadas: *Identification and authentication of integrated circuits.* Concurrency and Computation: Practice & Experience, pp. 1077 - 1098, Volume 16, Issue 11, September 2004.
12. P. Tuyls, B. Skoric: *Strong Authentication with Physical Unclonable Functions.* In: Security, Privacy and Trust in Modern Data Management, M. Petkovic, W. Jonker (Eds.), Springer, 2007.
13. G. Edward Suh, Srinivas Devadas: *Physical Unclonable Functions for Device Authentication and Secret Key Generation.* DAC 2007: 9-14
14. B. Gassend, M. v. Dijk, D. E. Clarke, E. Torlak, S. Devadas, P. Tuyls: *Controlled physical random functions and applications.* ACM Trans. Inf. Syst. Secur. 10(4), 2008.
15. U. Rührmair: *Oblivious Transfer based on Physical Unclonable Functions (Extended Abstract).* TRUST 2010, Lecture Notes in Computer Science, Volume 6101, pp. 430-440, Springer, 2010.
16. U. Rührmair: *SIMPL Systems: On a Public Key Variant of Physical Unclonable Functions.* Cryptology ePrint Archive, Report 2009/255, 2009.
17. U. Rührmair, Q. Chen, P. Lugli, U. Schlichtmann, M. Stutzmann, G. Csaba: *Towards Electrical, Integrated Implementations of SIMPL Systems.* Cryptology ePrint Archive, Report 2009/278, 2009.
18. Q. Chen, G. Csaba, X. Ju, S.B. Natarajan, P. Lugli, M. Stutzmann, U. Schlichtmann, U. Rührmair: *Analog Circuits for Physical Cryptography.* 12th International Symposium on Integrated Circuits (ISIC2009), Singapore, December 14 - 16, 2009.
19. U. Rührmair, Q. Chen, M. Stutzmann, P. Lugli, U. Schlichtmann, G. Csaba: *Towards Electrical, Integrated Implementations of SIMPL Systems.* Workshop in Information Security Theory and Practice (WISTP 2010), Passau (Germany), 2010. LNCS, Volume 6033, pp. 277 - 292, Springer Verlag, Berlin, 2010.

20. Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, M. Stutzmann, U. Rührmair: *Circuit-based Approaches to SIMPL Systems.* Accepted by Journal of Circuits, Systems and Computers, 2010. To appear.
21. N. Beckmann, M. Potkonjak: *Hardware-Based Public-Key Cryptography with Public Physically Unclonable Functions.* Information Hiding 2009: 206-220.
22. U. Rührmair, H. Busch, S. Katzenbeisser: *Strong PUFs: Models, Constructions and Security Proofs.* To appear in A.-R. Sadeghi, D. Naccache (Editors): Towards Hardware Intrinsic Security: Foundation and Practice. Springer, 2010.
23. Blaise Gassend, *Physical Random Functions*, MSc Thesis, MIT, 2003.
24. M. Majzoobi, A. Elnably, F. Koushanfar, *FPGA Time-Bounded Unclonable Authentication*. Information Hiding Conference, pp. 1-16, LNCS 6387, 2010.
25. J. Guajardo, S. S. Kumar, G. J. Schrijen, P. Tuyls: *FPGA Intrinsic PUFs and Their Use for IP Protection*. CHES 2007: 63-80.
26. U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, J. Schmidhuber: *Modeling Attacks on Physical Unclonable Functions*. 17th ACM Conference on Computer and Communications Security, 2010. Previous versions available from Cryptology ePrint Archive, Report 251/2010.
27. S. Halevi, H. Krawczyk: *MMH: Software Message Authentication in the Gbit/Second Rates.* FSE 1997: 172-189
28. Gerald DeJean, Darko Kirovski: *RF-DNA: Radio-Frequency Certificates of Authenticity.* CHES 2007: 346-363.
29. Youry Kariakin: *Authentication of Articles*. Patent writing, WO/1997/024699, available from http://www.wipo.int/pctdb/en/wo.jsp?wo=1997024699, 1995.
30. D. Vijaywargi, D. Lewis, D. Kirovski: *Optical DNA*. Financial Cryptography and Data Security, pp. 222-229, 2009.
31. G. Hammouri, A. Dana, B. Sunar: *CDs Have Fingerprints Too*. CHES 2009: 348-362.
32. W. Diffie and M.E. Hellman: *New Directions in Cryptography*. IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, pp. 644654.
33. Andrew Chi-Chih Yao: *Classical physics and the Church-Turing Thesis*. Journal of the ACM 50(1), 100-105, 2003.
34. Scott Aaronson: *NP-complete Problems and Physical Reality.* Electronic Colloquium on Computational Complexity (ECCC), 026, 2005.
35. Peter W. Shor: *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*. SIAM J. Comput. 26(5): 1484-1509 (1997)
36. G. Csaba, X. Ju, Z. Ma, Q. Chen, W. Porod, J. Schmidhuber, U. Schlichtmann, P. Lugli, U. Rhrmair: *Application of Mismatched Cellular Nonlinear Networks for Physical Cryptography*. IEEE CNNA, 2010.
37. D. Lim: *Extracting Secret Keys from Integrated Circuits.* M.Sc. Thesis, MIT, 2004.
38. G. E. Suh, C. W. O'Donnell, I. Sachdev, and S. Devadas: *Design and Implementation of the AEGIS Single-Chip Secure Processor Using Physical Random Functions*. Proc. 32nd ISCA, New York, 2005.
39. Meng-Day (Mandel) Yu, Srinivas Devadas: *Secure and Robust Error Correction for Physical Unclonable Functions*. IEEE Design & Test of Computers 27(1): 48-65 (2010)
40. U. Rührmair, A. Weiershäuser, S. Urban, C. Hilgers, J. Finley: *Secure Integrated Optical Physical Unclonable Functions.* In preparation, 2010.
41. Stephen G. Lipson: *Optical Physics.* 3rd ed., Cambridge University Press, 1995. ISBN 0-5214-3631-1.
42. Wolfgang Demtröder: *Experimentalphysik 2: Elektrizität und Optik.* Springer 2004. ISBN-10: 3540202102.
43. D. Zhou, L.J. Mawst: *Two-dimensional phase-locked antiguided vertical-cavity surface-emitting laser arrays*. Applied Physics Letters, 2000.

A A First Proof-of-Concept for Optical SIMPLs

In order to rigorously prove the validity of the suggested optical SIMPL implementation, two statements would have to be shown. (i) The system indeed has the desired speed advantage. (ii) Our suggestion is workable in the sense that its responses can be predicted sufficiently accurately by the described approach. Similar to the security of classical cryptoschemes, statement (i) cannot be shown or proven mathematically in a strict sense given the current state of computational complexity theory. Also building an optical SIMPL prototype that operates at optimized operational speed is expensive and beyond the scope of this paper.

Nevertheless, it proved well doable to build a prototype that is not optimized in terms of speed, but which verifies statement (ii). It occured that such a prototype had been already set up in our group in the course of a different study, where we generally investigated the machine learnability of integrated optical PUFs [40]. We found there that it was indeed possible to machine learn the output of linear optical PUFs with high accuracy. This has direct implications for the realizability of optical SIMPL systems; we quote from the work [40] in the sequel.

The set-up we used in [40] is depicted schematically in Figure 2. It consists of a LCD array from an old beamer acquired via ebay for 20 Euros, several lenses (depicted schematically in one symbol) and a scattering token of small glasspheres and a transparent glue ("UHU Schnellfest"). A pattern is switched on in the LCD array, and the laser is directed towards it. The set-up has the same effective operational functionality as a array of phase-locked laser diodes, but was easier to realize with components present in our laboratory.

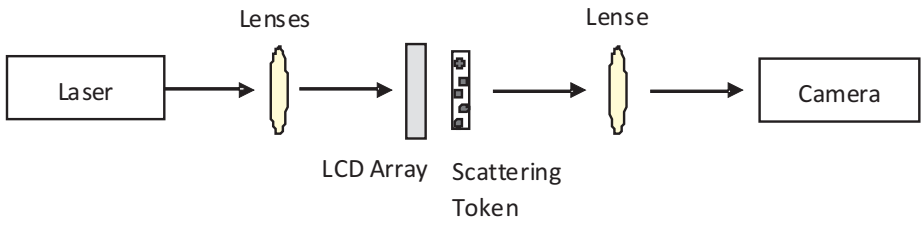

Fig. 2. Schematic illustration of our measurement set-up. The schematic icon "lenses" on the left stands for several lenses that were used to shape the light beam.

We applied the method described in Section 4.3 in order to predict the outcome of the speckle pattern. We found that the predictability on the basis of optical wave superposition works not only well in theory, but also in practice. Standard ML regression were applied to 53,700 different CRPs (i.e. patterns on the LCD array and corresponding CCD images) that were collected. The success for two different excitation patterns in shown in Figures 3 and 4. The difference map between the actually acquired optical

image and the prediction is contained in the figures, and is very small compared to the natural fluctuations in optical speckle patterns, for example due to laser fluctuations, which were already reported in [8] [9]. The variations observed in the difference map will presumably likely not be noticeable after the usual image transformations have been applied to the output. This illustrates the basic feasibility of predicting the output of optical SIMPLs.

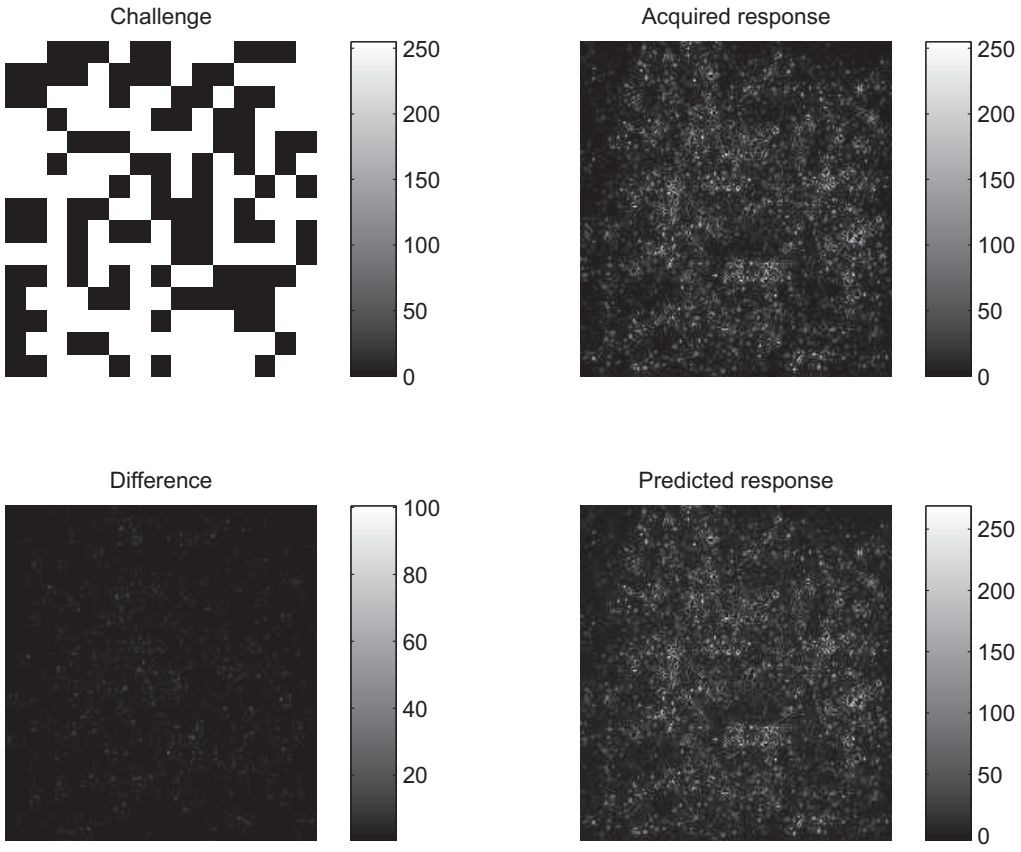

Fig. 3. A randomly chosen 15×15 excitation pattern (top left), a CCD image of the response of the optical SIMPL (top right), the predicted response (bottom right), and the difference map (bottom left).

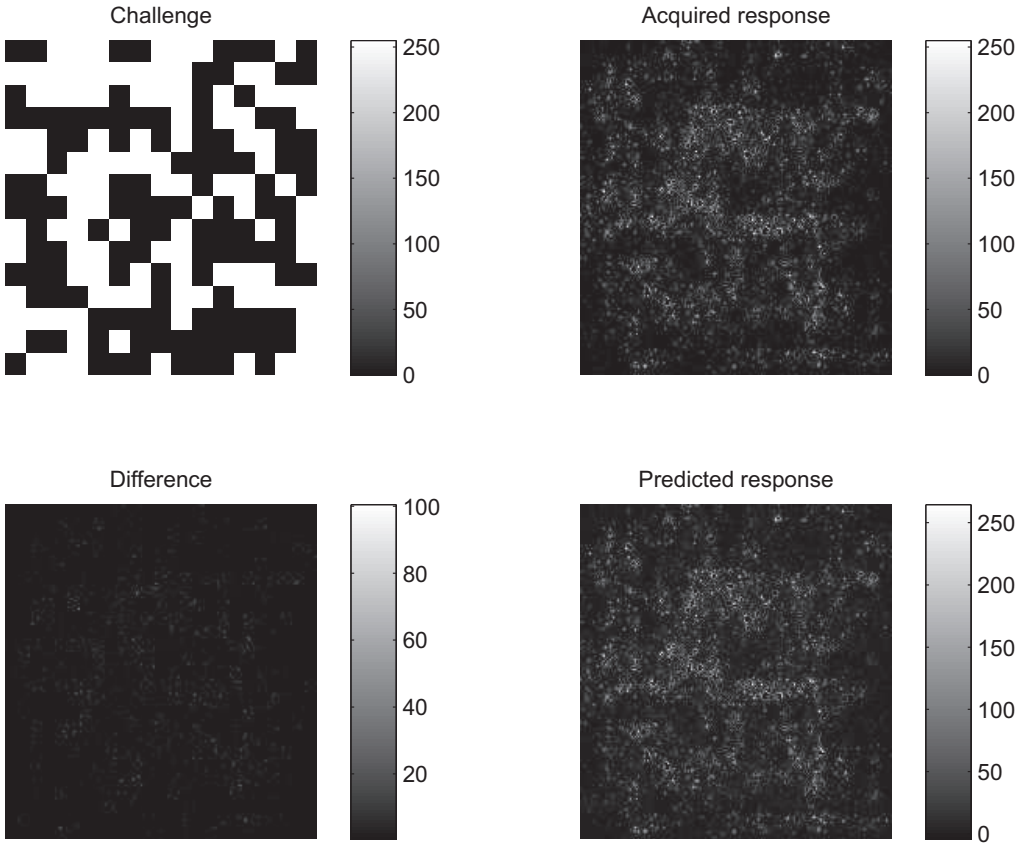

Fig. 4. A second, randomly chosen 15×15 excitation pattern (top left), a CCD image of the response of the optical SIMPL (top right), the predicted response (bottom right), and the difference map (bottom left).