Hardware Trojan Side-Channels Based on Physical Unclonable Functions

Zheng Gong[1,⋆] and Marc X. Makkes[2]

[1] School of Computer Science, South China Normal University
Guangzhou, 510631, China
cis.gong@gmail.com
[2] Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
m.x.makkes@kr85.org

Abstract. The separation design and fabrication process in the semiconductor industry leads to potential threats such as trojan side-channels (TSCs). In this paper we design a new family of TSCs from physical unclonable functions (PUFs). In particular, a dedicated attack on the PRESENT block cipher is described by using our PUF-based TSCs. Finally we analyze the performance of our PUF-based TSCs and discuss other potential applications.

1 Introduction

With the rapid developments of semiconductor technology, integrated circuits (ICs) are fast becoming an overwhelming presence in our daily lives. Since information security attracts more and more concerns, security chips are widely used to provide hardware support of cryptographic algorithms and obtain trust computing bases. In most cases, it becomes theoretically infeasible to directly attack a well-analyzed cryptographic algorithm (e.g., AES) within a security chip by using traditional cryptanalysis. Although a security chip can resist the attacks at the algorithm level, the weaknesses in the implementation level might be analyzed for practical attacks. For instance, the side-channel information, such as differential time or power analysis, is widely investigated to break the security protection of embedded systems [11,13].

The original ideas of trojan side channel attacks and covert channels were first proposed by Simmons [15]. As an aggressive example of side-channel attacks, Lin *et al.* [7] introduced the concept of *trojan side channels* (TSCs). A TSC can be viewed as a malicious circuit that can compromise information from an embedded crypto core, afterwards it can send out the information via side-channel signals. Only the attacker who implements the TSC can decode the information. Although TSCs require extra hardware costs, it is hard to detect since

⋆ This author's research was mainly performed while at DIES group, University of Twente, The Netherlands.

C.A. Ardagna and J. Zhou (Eds.): WISTP 2011, LNCS 6633, pp. 294–303, 2011.

they usually occupy a negligible amount of area in the genuine IC. The current IC supply chain, such as outsourced manufacturing, also provides great opportunities to implant malicious circuits into the genuine IC to compromise their security. Nevertheless, many governments and agencies require that companies who use encrypted communications systems (e.g., mail services from Blackberry) to allow these institutions to recover encrypted information with a feasible effort. Normally this requirement leads to two options for vendors: either choose key escrow or weak design of cryptography. Implementing TSCs can match this requirement without relying on the above options.

On the other side, detecting flaws in the lithography process is usually done with extra hardware supports and it is often used to check if the functionality is correct. Verification of the functionality is often with some extra hardware attached to the IC. Recent developments showed that it is possible for an attacker to modify chip designs and add malicious circuits without changing the functionality. In the literature, many approaches have been proposed for trojan hardware detection, such as visual inspections, test patterns to find unexpected behavior, side-channel and path delay profiles [3]. Currently, it is still infeasible to detect a large amount of security chips whether they have been affected by a trojan hardware with very small gate counts. In order to keep the TSC undetectable, it is also crucial to blind the side-channel information for other parties except the original attacker. In [7], Lin *et al.* suggest to use a LFSR for encoding. This results a practical problem that every chip implemented with the same LFSR will output the same stream. Therefore, if anyone resolves the polynomial that constructs the LFSR, it will be straightforward to decode the information from every TSC based on this LFSR.

In 2001, Pappu *et al.* [12] introduced the concept of *physical unclonable functions* (PUFs, also known as physical random functions). Since it is practically impossible to model, copy, or control the IC manufacturing process variations, PUFs can make chips unique and effectively unclonable. In this paper we propose a new familty of TSCs based on PUFs. The advantages of using a PUF-based TSC are two-fold: 1) for every TSC, it is unnecessary to be implemented with different LFSRs (or keys) but a PUF with the same circuits to blind its side-channel information. The one-wayness of the PUF protects the side-channel information can only be decoded by the attacker who implemented the TSC. 2) An attacker can trace the side-channel information from a certain chip by using the physical unclonable property of PUFs. It also means mathematically modeling one chip for recovery will be useless to other chips. We propose a PUF-based TSC attack on the PRESENT block cipher to show the relatively negligible hardware implementation cost compared to genuine ICs.

The remainder of the paper is organized as follows: Section 2 reviews the preliminaries for TSCs and PUFs, Section 3 first describes a generalized model for PUF-based TSC attacks, after which propose a PUF-based TSC attack on the PRESENT block cipher, Section 4 discusses other applications of PUF-based TSCs, and Section 5 concludes the paper.

2 Preliminaries

2.1 The Trojan Side-Channel Model

It is widely accepted that a well-defined model must be formalized on the system that requires analysis. In [7], Lin *et al.* introduces the parties and activities that are involved with the Trojan Side-Channel (TSC) model. Here we will refine the TSC model from a more general perspective of the TSC scenario.

Entities & Activities. A TSC can be used either by a malicious attacker or by an anti-counterfeiting analyzer. Without loss of generality, we call the party who implants the Trojan hardware into the circuits *tracer* $\mathcal{T}$, and the party who attempts to detect those TSCs *evaluator* $\mathcal{E}$. For malicious applications, $\mathcal{T}$ will try to hide the usage of TSCs, while $\mathcal{E}$ will try to verify the correctness and integrity of circuits. For anti-counterfeiting usages, $\mathcal{T}$ will try to expose the side-channel information of TSCs, while $\mathcal{E}$ will try to discover and hinder the leakage by TSCs. Except for the implanted TSCs, we assume that the genuine ICs are tamper-resistant and no other side-channels can be found by $\mathcal{T}$. $\mathcal{E}$ can extensively test the functionality of the genuine ICs and capture the signals leaked out by TSCs.

Requirements. To evaluate its implementation quality, a TSC must obey the following conditions.

- *Circuit properties*:
  - Imperceptibility. Compared to the genuine IC, a TSC must only increase a negligible area of logic gates to reduce the possibility of detections by evaluators.
  - Conformity. A TSC must not affect the correctness and integrity of the genuine ICs. Moreover, the timing properties of the genuine ICs (e.g., cycles for an encryption/decryption) will not be affected overtly by an implanted TSC.
- *Signal properties*:
  - Blindness. Except for tracers who implanted the TSC, side-channel information leaked out by a TSC must be blind to other parties. That is to say, evaluators cannot distinguish the difference between information leaked out by TSCs and bits from a pseudorandom number generator.
  - Latency. To avoid the detection, a TSC will be latent unless it is triggered with a certain condition predefined by tracers. The trigger must be imperceptible from extensive functionality testing of the genuine ICs.

2.2 Physical Unclonable Functions

It is known that PUFs exploit the physical characteristics of the silicon and the IC manufacturing process variations to uniquely characterize each and every silicon chip. The unclonability property comes from the fact that a PUF consists

of a finite number of random components, which is infeasible to exactly control over the manufacturing process. Each PUF uses these random components to map *challenges* to *responses* (CRPs). A challenge is a stimulus that is applied to the PUF and a response is the reaction of the PUF obtained through measurements. Due to the complex interaction of the stimulus with the physical microstructure of the device, each PUF will trigger a response that is highly unpredictable and unique. PUFs are often used to setup secure channels between devices. At the manufacturing process the manufacturer creates a set of CRPs with a PUF and hands them over to user. Therefore, a user can set a secure channel using the CRP by sending a challenge C to the PUF. Since the response R of the PUF is also known to the user, it can be used as a shared secret key for secure communication.

In the literature, many types of PUFs have been proposed based on different physical properties. By using the position of light as the challenge, Pappu *et al.* [12] proposed a PUF based on the scattering of light when shining a laser on a bubble-filled transparent epoxy wafer. In [6], Lim *et al.* introduced a new family of PUFs based on arbiters (APUF). An APUF consists of two identically configured delay paths that includes a number of switches. The switches are set by the stimulus and determine a unique path that signals have to travel. In order to generate response bits for an APUF, a signal is activated simultaneously on both delay paths. At the end of the delay paths there is an edge triggered flip-flop which determines the fastest signal by outputting a signal bit.

3 PUF-Based TSC Attacks

3.1 A Paradigm on PRESENT

The key objective of a TSC is to compromise key information from a crypto engine without altering or delaying the process of the genuine IC. Altering or delaying might reveal the existence of the TSC to evaluators. When a tracer $\mathcal{T}$ exploited the side-channel information, $\mathcal{T}$ will first decode the information to the original message. By retrieving the saved CRPs of the implanted PUFs, $\mathcal{T}$ can identify which IC leaked out the side-channel information, and then recover the compromised key information. In [7], Lin *et al.* adapted the concepts from spreads-spectrum communications (also known as code-division multiple access (CDMA)) to distribute the compromised bits as a covert channel to $\mathcal{T}$. For simplicity, we also choose it as the leakage circuit for PUF-based TSCs. Figure 1 depicts a generalized model for PUF-based TSC attacks.

Since TSCs also require signal blindness, traditional PUFs are not resource-efficient for ensuring the randomness of long length CRPs. Here we propose a new variant of PUF which is suitable for TSCs. Basically, our design (see Figure 2) is constructed from a combination of a *feedback shift register* (FSR) and a number of APUFs. The FSR can store a bit string with the length C and shift P bits in every cycle. Each APUF consists of C switch elements, and the number of APUFs is P. The FSR provides each APUF with an identical challenge, as the output of the APUFs is inserted back in the FSR and is sent out over the

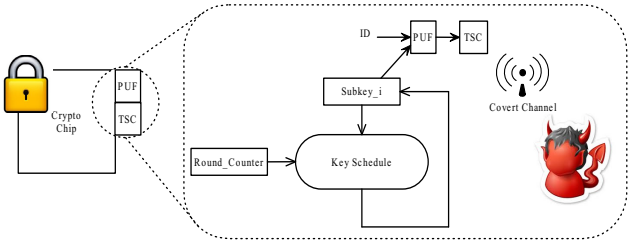

Fig. 1. PUF-based TSC attacks

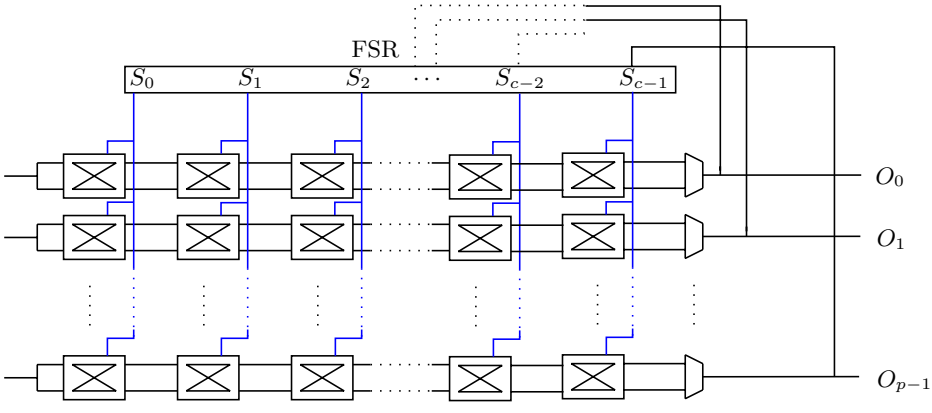

Fig. 2. A new variant of TSC based on PUFs. The top rectangle is a c-bit shift-register. The register is shifted by p-bits per cycle. Each bit is tapped and used to set the set the switch components of the PUFs. The output O_i produced by the PUFs are feedback to the shift-register and XORed with the keybits.

covert channel. We denote that a PUF-based TSC with only a single APUF to be a serial PUF-based TSC.

Initially, the FSR is loaded with a unique identifier (ID). To prevent the detection caused by the power usage, the PUF-based TSC processes and transmits key information (Key) at the same time as encryption occurs. In order to transmit the side-channel information, we suggest the following format for the covert channel.

$$\text{PUF(ID)}||\text{PUF(Key)}||\cdots\text{PUF(ID)}||\text{PUF(Key)}\cdots$$

Using this format $\mathcal{T}$ can first decode the signal and retrieve the PUF(ID) which can be looked up in the list of devices. Afterwards, $\mathcal{T}$ can decode the PUF(Key) because the CRPs of each APUF are stored (to a database) beforehand. The length of an APUF (i.e., the number of switches) determines the storage requirement per device.

For a PUF-based TSC, the security and performance of a given structure determine its area costs. The first parameter is the length of the APUF which regulates the security of the structure. The second parameter is the number of APUFs which determine the performance of a PUF-based TSC by clock cycles. The gate equivalents (GE) of the PUF-based TSC also relies on the implementation of the cryptosystem. In [10], Ozturk *et al.* showed that a PUF with a 64-bit challenge and a single bit response using a tri-state APUF can be implemented in 351 gates. We will consider smaller challenges in our PUF-based TSCs for the imperceptibility.

PRESENT. At CHES 2007, Bogdanov *et al.* proposed an ultra-lightweight block cipher which is named PRESENT [2]. PRESENT is an example of an SP-network and consists of 31 rounds. The block length is 64 bits and two key lengths of 80 and 128 bits are supported. The hardware requirements for PRESENT are competitive. Using the *Virtual Silicon* (VST) standard cell library based on *UMC L180 0.18μm 1P6M Logic Process* (UMCL18G212T3), PRESENT-80 and PRESENT-128 are estimated to require 1570 and 1886 gate equivalents, respectively [2]. Since Bogdanov *et al.* do not expect the 128-bit key version to be used until a rigorous analysis is given, the term PRESENT means the 80-bit key version in hereafter. A high-level algorithm of the round function of PRESENT is depicted in Figure 3.

$generateRoundKeys(k) \rightarrow \{k_1, k_2, \cdots, k_{32}\}$;
for $i = 1$ *to* 31do
  $addRoundKey(\text{STATE}, k_i)$;
  $sBoxLayer(\text{STATE})$;
  $pLayer(\text{STATE})$;
end for
$addRoundKey(\text{STATE}, k_{32})$.

Fig. 3. The round function of PRESENT

The key schedule. PRESENT uses a hardware-efficient key schedule to avoid the scheduling weaknesses, which may be used for the related-key attack and the slide attack. The user-supplied key is stored in a key register K and represented as $k_{79}k_{78}\cdots k_0$. At the i-th round, the leftmost 64-bit of the current key register becomes the subkey $K_i = k_{79}k_{78}\cdots k_{16}$. Subsequently, the key register K is updated as follows.

- Cycling left shift 61 bits such that $[k_{79}k_{78}\cdots k_0] = [k_{18}k_{17}\cdots k_{20}k_{19}]$,
- The leftmost 4 bits are passed through the PRESENT S-box such that $[k_{79}k_{78}k_{77}k_{76}] = S[k_{79}k_{78}k_{77}k_{76}]$,
- The round counter value is XORed with bits $k_{19}k_{18}k_{17}k_{16}k_{15}$.

Based on the key schedule of PRESENT, we can design a lightweight TSC as follows. For each round, TSC will try to leak out the rightmost 4 bits of K. Since

the key register is cycling left shift 61 bits, the rightmost 4 bits will not repeat themselves within 21 rounds (as illustrated in Figure 4). Thus we can obtain the whole key bits after 21 rounds of PRESENT. Since only the leftmost 64 bits in the key register will be used in the addRoundKey algorithm, compromising the rightmost 4 bits will not imply any unexpected behavior or path delay on PRESENT.

$$
\begin{aligned}
\textit{Round } 1 &: k_{79}k_{78}k_{77}k_{76} \cdots \underline{k_3k_2k_1k_0} \\
\textit{Round } 2 &: k_{18}k_{17}k_{16}k_{15} \cdots \underline{k_{22}k_{21}k_{20}k_{19}} \\
\textit{Round } 3 &: k_{37}k_{36}k_{35}k_{34} \cdots \underline{k_{41}k_{40}k_{39}k_{38}} \\
\textit{Round } 4 &: k_{56}k_{55}k_{54}k_{53} \cdots \underline{k_{60}k_{59}k_{58}k_{57}} \\
&\cdots \\
\textit{Round } 11 &: k_{29}k_{28}k_{27}k_{26} \cdots \underline{k_{33}k_{32}k_{31}k_{30}} \\
&\cdots \\
\textit{Round } 21 &: k_{59}k_{58}k_{57}k_{56} \cdots \underline{k_{63}k_{62}k_{61}k_{60}}
\end{aligned}
$$

Fig. 4. Key Scheduling of PRESENT

Instead of recovering a full-length key, a parameterized approach can be used to lower the length of sending bits via side channels. Similar to the above attack on PRESENT, one can carefully choose a combination of key bits and rounds for a TSC. For instance, we can make a TSC that leaks 4 bits in each round and stops after 11 rounds. After 11 rounds of PRESENT, we can obtain 40 bits of the original key and execute an exhaustive search for the rest of 40 bits key. The exhaustive search only requires a time complexity of about $O(2^{40})$, which can be executed in minutes on current PCs. Consequently, a PUF-based TSC attack on PRESENT can be implemented as follows.

1. Send a pre-distributed identifier ID as the challenge to the PUF, obtain the response $r_1 = \text{PUF(ID)}$.
2. Eavesdrop each of the rightmost 4 bits of 11 rounds subkeys from the underlying PRESENT encryption/decryption. Input the compromised bits C to PUF, obtain the response $r_2 = \text{PUF}(C)$.
3. Encode the complete message $R = r_1||r_2$, which will be sent by the covert channel.

Although the key scheduling algorithm of PRESENT has no security problems so far under cryptanalysis, our proposed attack endangers its security in practical implementations. We note that the above attack on PRESENT can also be extended to other ciphers designed with a "simple" key scheduling algorithm. If a key scheduling algorithm has a low non-linear complexity (i.e., mainly relies on linear operations such as bit shifts), a TSC attacker can easily recover the entire secret key by eavesdropping a few bits of the subkey in each round. If TSC attacks are considered in the adversary model, the non-linearity of the key schedule should be carefully strengthened by algorithm designers.

3.2 Performance Analysis

To estimate the lower-bound GE of our PUF-based TSC, we take the formula $P \cdot (5 \cdot C + 4) + 4 \cdot C)$ where C is the length of the challenge and P is the number of the APUFs. Our lower-bound GE estimation is based on the implementation of PUF-based TSC in *Xilinx ISE Design Suite 12.2*. If we consider a 24-bit challenge for the PRESENT implementation where 4 key bits are snooped per cycle, the lower-bound of GE of the PUF-based TSC can be 592 gates. It is possible to lower the GE by lowering the number of APUFs. Table 1 gives an overview of the performance of implementations that are derived from different parameters of the PUF-based TSC. The results start with a low-area and low-performance implementation (serial PUF-based TSC), to a high-area and high performance implementation (TSC with 4 APUFs). It is obvious from the results that performance comes at a cost in area and storage while the length of APUF only significantly results in storage requirements.

Table 1. The performance our PUF-based TSCs

	TSC width (bit)	FSR length (bit)	CRPs storage	Performance in cycles	Area in GE
Serial PUF-based TSC	1	24	16MB	68	220
TSC with 2 APUFs	2	24	32MB	34	344
TSC with 4 APUFs	4	24	64MB	17	592
Serial PUF-based TSC	1	36	420MB	80	328
TSC with 2 APUFs	2	36	840MB	40	512
TSC with 4 APUFs	4	36	1.68GB	20	880

3.3 Evaluation

The Imperceptibility of a PUF-based TSC heavily depends on the size of the genuine chip. If a chip only contains gates nearly or below a thousand level, the TSC can be easily spotted. Our proposed TSC requires 592 gates, where as PRESENT requires 1570-1886 gates. This is almost 30-40% of the area of the attacked cipher. In [1], Agrawal *et al.* comments the that hardware trojans are detectable if the size of the trojan is more than 0.01% of the floorplan. But if a chip has thousands of gates, or a million chips need to be examined, spotting the TSC becomes increasingly difficult. Moreover, we recognize that it is a possible and clever way to hide this type of TSC in a protection mesh, such as in smartcards. In [14] Ruhrmair *et al.* presented a modeling attack on PUFs by using *linear regression* to a mapping a large amount of challenges to responses and deriving a model of the PUF. But in our PUF-based TSC attacks, the challenge for the PUF is derived form the round key. For an attacker to construct an key such that specific bits are set in the round key, it is infeasible to build a modeling attack on such CRPs of a PUF.

Besides that PUFs are well known and easily implementable in FPGAs [8,9], Devadas *et al.* showed in [5] how PUFs can be practically implemented in ASICs. Since our PUF-based TSC heavily relies on the implementation of PUFs, it can

also be feasibly implemented even on a large scale. Although its feasibility and imperceptibility still require deeper investigation, our PUF-based TSC has many advantage over the TSC presented by Lin *et al.* [7]. Firstly, our PUF-based TSC can provide unpredictable outputs that are related to physical unclonability, which increases the blindness and the uniqueness of side-channel information. Secondly, PUF-based TSCs can be parameterized by the consideration of performance and resource limitations. This gives a high level of adaptability as shown with the attack on PRESENT. Note that our PUF-based TSC attack on PRESENT can be extended for other block ciphers with a similar bit-shifting key schedule.

Although Lin *et al.* [7] have not described how to activate the TSC in their proposal, here we provide a possible design for triggering the PUF-based TSC. Since PRESENT has a round counter for 31 rounds key schedule, while our proposed attack only requires the leakage bits of 11 rounds, a possible trigger can start the PUF-based TSC between $0 \leq i < 20$ where i is the round counter. The number i is variable and can be selected attacker at the manufacturing stage. This trigger requires some additional administration (i.e., 4 bits per PUF-based TSC) as the attacker needs to know which key bits of which round are sent.

4 Other Applications

Except direct attacks on key recovery, PUF-based TSCs can be used in many other applications. Two interesting examples are described as follows.

Personal communication eavesdropping. Nowadays, mobile devices are widely secured with cryptographic algorithms. If a PUF-based TSC is implanted to a certain user's device, a tracer can not only eavesdrop side-channel information, but can also identify such signals sent from a certain device that belongs to a certain user.

Parameterized backdoor. Some countries have restrictive regulations on the exporting of security chips, which impose that only chips lower than a certain security level can be shipped. Our parameterized TSC on PRESENT shows it can also be used as a factor to lower the security level of a tamper-resistant chip and therefore matches those exporting limitations.

5 Conclusion

In this paper we introduced a new type of flexible TSC based on PUFs. A dedicated PUF-based TSC has been proposed for attacking PRESENT. Compared to Lin et al.'s original proposal, our PUF-based TSCs cleverly uses the physical unclonability to obtain the blindness of side-channel information. The implementation results support that PUF-based TSCs can also be lightweight in logic gates, which is an important factor for the imperceptibility of TSC circuits. In future, we are interested in designing PUF-based TSCs on other cryptographic primitives that are practically used in security chips.

Acknowledgement. We would like to thank many anonymous reviewers for their valuable comments. The first author is supported by NSFC (No.6170217), National "863" Program of China (No. 2009AA01Z418) and National "973" Program of China (No.2007CB311201).

References

1. Agrawal, D., Baktir, S., Karakoyunlu, D., Rohatgi, P., Sunar, B.: Trojan detection using IC fingerprinting. In: IEEE Symposium on Security and Privacy, pp. 296–310. IEEE Computer Society, Los Alamitos (2007)
2. Bogdanov, A.A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: An ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007)
3. Chakraborty, R.S., Wolff, F.G., Paul, S., Papachristou, C.A., Bhunia, S.: Mero: A statistical approach for hardware trojan detection. In: Clavier and Gaj [4], pp. 396–410
4. Clavier, C., Gaj, K. (eds.): CHES 2009. LNCS, vol. 5747. Springer, Heidelberg (2009)
5. Devadas, S., Suh, E., Paral, S., Sowell, R., Ziola, T., Khandelwal, V.: Design and implementation of PUF-based unclonableRFID ICs for anti-counterfeiting and security applications. In: IEEE International Conference on RFID, pp. 58–64 (2008)
6. Lim, D., Lee, J.W., Gassend, B., Edward Suh, G., van Dijk, M., Devadas, S.: Extracting secret keys from integrated circuits. IEEE Trans. VLSI Syst. 13(10), 1200–1205 (2005)
7. Lin, L., Kasper, M., Güneysu, T., Paar, C., Burleson, W.: Trojan side-channels: Lightweight hardware trojans through side-channel engineering. In: Clavier and Gaj [4], pp. 382–395
8. Merli, D., Stumpf, F., Eckert, C.: Improving the quality of ring oscillator pufs on fpgas. In: WESS, p. 9. ACM, New York (2010)
9. Morozov, S., Maiti, A., Schaumont, P.: An Analysis of Delay Based PUF Implementations on FPGA. In: Sirisuk, P., Morgan, F., El-Ghazawi, T., Amano, H. (eds.) ARC 2010. LNCS, vol. 5992, pp. 382–387. Springer, Heidelberg (2010)
10. Ozturk, E., Hammouri, G., Sunar, B.: Physical unclonable function with tristate buffers. In: IEEE International Symposium on Circuits and Systems, pp. 3194–3197. IEEE, Los Alamitos (2008)
11. Paar, C., Eisenbarth, T., Kasper, M., Kasper, T., Moradi, A.: Keeloq and side-channel analysis-evolution of an attack. In: Breveglieri, L., Gueron, S., Koren, I., Naccache, D., Seifert, J.-P. (eds.) Sixth International Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2009, Lausanne, Switzerland, September 6, pp. 65–69. IEEE Computer Society, Los Alamitos (2009)
12. Pappu, R., Recht, B., Taylor, J., Gershenfeld, N.: Physical One-Way Functions. Science 297(5589), 2026–2030 (2002)
13. Popp, T., Kirschbaum, M., Mangard, S.: Practical attacks on masked hardware. In: Fischlin, M. (ed.) CT-RSA 2009. LNCS, vol. 5473, pp. 211–225. Springer, Heidelberg (2009)
14. Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., Schmidhuber, J.: Modeling attacks on physical unclonable functions. In: Al-Shaer, E., Keromytis, A.D., Shmatikov, V. (eds.) ACM Conference on Computer and Communications Security, pp. 237–249. ACM, New York (2010)
15. Simmons, G.J.: The prisoners' problem and the subliminal channel. In: CRYPTO, pp. 51–67 (1983)