A New Remote Keyless Entry System Resistant to Power Analysis Attacks

Amir Moradi and Timo Kasper
Horst Görtz Institute for IT Security, Ruhr University Bochum, Germany
Email: {moradi, tkasper}@crypto.rub.de

***Abstract*—In CRYPTO 2008, it has been shown that power-analysis attacks can completely break real-word remote keyless entry (RKE) systems based on the KEELOQ code-hopping scheme. A successful key-recovery involves a couple of security and privacy risks for the consumers as well as for the manufacturers. In this paper, we introduce a new RKE system that is inherently resistant against side-channel attacks, independent of the implementation platform. For our approach, a pseudo random number generator (PRNG) as introduced in ASIACCS 2008 is extended to prevent template attacks, and embedded into a secure remote-control application. We verify the effectiveness of the proposed scheme by implementing it on a microcontroller and evaluate its physical security, thereby demonstrating that a practical key-recovery by means of power-analysis is not feasible.**

I. Introduction

A. History

The first generation of RKE systems allow for controlling the access to a secured object by sending a fixed sequence of data from the remote control to the receiver [9]. This unidirectional scheme is, contrary to a bidirectional challenge-and-response authentication, obviously at risk of a replay attack. To protect against the latter, in 1995 Microchip Technology Inc embedded the so-called code-hopping mechanism, based on the proprietary block cipher KEELOQ. A KEELOQ remote control possesses an internal counter that is increased and encrypted each time a hopping code (also called rolling code) is sent [12]. A KEELOQ receiver validates a hopping code only if the decrypted value of the counter is moderately increased in comparison with the last valid one, hence replay attacks can be prevented by rejecting repetitive codes.

Since its recent cryptanalysis [1], [4], [7], the KEELOQ block cipher has to be regarded as insecure from the cryptographic point of view. The mathematical attacks rely on known pairs of plain- and ciphertexts and are hence only practical for KEELOQ challenge-and-response protocols, e.g., [13]. Due to the fact that plaintexts are not accessible to an attacker in the case of a code-hopping scheme, the corresponding RKE systems were still regarded as secure, until to the publication of a practical key-recovery by means of power-analysis in CRYPTO 2008 [5]. The authors describe severe attacks on KEELOQ RKE systems, implying a complete break of all products of a manufacturer after extracting his secret key by means of power-analysis. The knowledge of the manufacturer key, which can be recovered from only one single power measurement of the software running in any receiver [8], allows for taking over control of all RKE systems of that manufacturer just by eavesdropping a transmission from a remote control to the receiver [5]. This single point-of-failure in the design of the KEELOQ crypto-system violates the privacy and security of the customers, as the manufacturer of a KEELOQ code hopping system (or criminals gaining his manufacturer key illegally) can unnoticedly access any secured object where the system is installed.

B. Related Work

Several schemes have been proposed to protect cryptographic implementations with respect to side-channel analysis, particularly to differential power-analysis (DPA). The goal of DPA countermeasures is to avoid a dependency between the power consumption of a cryptographic device and intermediate values of the executed algorithm [10]. *Hiding* and *Masking* are amongst the most common countermeasures on either the hardware or the software level. *Hiding* methods aim at increasing the noise factor [22] or at equalizing the power consumption values [20] independently of the processed data, thereby decreasing the signal-to-noise ratio (SNR). *Masking* countermeasures rely on randomizing all key-dependent intermediate values processed during the execution of the cipher and can be employed on either algorithmic level [16] or cell level [18], often combined with *shuffling* [6], i.e., randomizing the order of the operations, and inserting *dummy rounds* that appear like genuine rounds of the cipher but process no relevant information. However, each of these methods comes along with some costly disadvantages and implementation constraints, e.g., increased power consumption, enlarged chip-area, and increased execution time. Despite of the increased efforts required by an adversary, none of the so far proposed techniques can perfectly counteract a key-recovery by means of DPA in practice and improved attacks can defeat even sophisticated countermeasures [3], [11], [21].

A new approach to thwart side-channel analysis has been presented in ASIACCS 2008 [17]. The authors propose a re-keying strategy via a PRNG based on a block cipher and theoretically investigate the resulting side-channel leakage of the implementation. The basic idea is that each session key is used only once, hence those side-channel attacks requiring to combine several queries in order to reach high success rates, like DPA, become ineffective. Still, this method cannot perfectly hinder the most powerful key-recovery attack based on side-channel analysis, namely template attack [2], that

978–1–4244–4657–5/09/$25.00 © 2009 IEEE

ideally requires just a single leakage query. The PRNG should therefore be combined with other countermeasures such as hiding to counteract template attacks more efficiently.

C. Motivation

The vulnerability of KeeLoq RKE systems and the aforementioned secure PRNG inspired us to develop a new RKE scheme that is inherently resistant to power-analysis attacks and can provide reliable security and privacy to the customers. Since the PRNG shall serve as a DPA-resistant core of our proposal, and the platform of RKE systems is usually a general-purpose 8-bit microcontroller, the implementation would be vulnerable to template attacks. Accordingly, we intend to modify the design of the PRNG in order to achieve a reasonable resistance against profiling attacks. To ensure that the achieved performance is suitable for commercial systems and to verify the resistance against DPA and template attacks, we will practically implement and analyze our proposed system on an 8-bit microcontroller.

D. Organization

In the following, an introduction of the operating principle of rolling-code schemes and an analysis of their security vulnerabilities are given in Section II. Our new RKE system is then presented in Section III, including a description of its functionality and details about sharing its secret parameters. Section IV investigates the security of our proposals from the theoretical point of view and illustrates the final architecture emerging from the physical security analyses. As a proof-of-concept, we implement and test the developed RKE application on a microcontroller in Section V and address difficulties and limitations in the context of real-world scenarios.

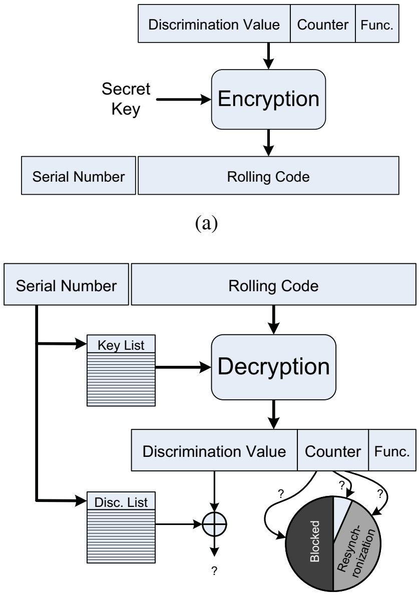
II. Rolling Codes

A. Structure

The goal of a rolling code system, as depicted in Fig. 1(a), is to generate dynamic transmissions with a remote control that contain the commands to be executed by a receiver, in order to defeat replay attacks in a unidirectional system. In a typical scheme, a remote control possesses a unique and public "serial number", a fixed "discrimination value", an internal "counter", and a "secret key". Each time one of its buttons is pressed the counter in the remote is increased, concatenated with the discrimination value and the command to be executed, and then encrypted with the secret key. The resulting ciphertext, called rolling code, is appended with the serial number of the remote before being sent to the receiver.

The receiver, which has stored the serial number, the discrimination value, the last counter value, and of course the secret key of each valid remote control, updates its respective counter and permits access to the secured object only if the following three criteria are satisfied:

1) The serial number of the received rolling code must be valid, i.e., the corresponding remote including its secret key must be known by the receiver.
2) The discrimination value of the decrypted rolling code must be identical to the stored one.
3) The counter value contained in the decrypted rolling code must be moderately increased compared to the last one.



(a)

(b)

Fig. 1. Rolling code scheme (a) in the transmitter and (b) in the receiver

In order to prevent from circumventing the RKE system by sending random counter values, as well as from replay attacks, the counter space[1] is split up into the three windows illustrated in Fig. 1(b). The first window is often small, containing no more than 16 values, and begins after the last valid received value of the counter. A remote control will be authenticated directly, if the received counter is within this window. The second so-called "resynchronization" window comprises the next codes typically up to half of the counter space. To authenticate the remote and resynchronize with its current counter value, the receiver must detect two consecutive counter values within this window. Any transmission whose counter value is within the third "blocked" window, consisting of the remainder of the counter space, will be ignored to preclude the repetition of previously monitored rolling codes.

Microchip Technology Inc proposes to employ the KeeLoq cipher for such a rolling code system. Since KeeLoq processes blocks of 32-bit values with a 64-bit secret key, the rolling code has a length of 32 bits. Consequently discrimination value, counter, and function index have a length of 12, 16, and 4 bits, respectively. Often, the transmitters use a cost-effective hardware implementation, e.g., [12], and the receivers perform the algorithm in software on 8-bit PIC microcontrollers, e.g., [14].

[1] Usually, the minimum space of the counter value is 16 bits, hence the same counter value reoccurrs after 2^{16} messages.

B. Key Management

In order to avoid recovering the secret key of all transmitters of a company by revealing only one, it is strongly recommended to generate a unique secret key for each remote control.

A receiver must be able to learn the secret key of new remote controls, hence an appropriate learning mechanism is required. It is out of bounds to send secrets in clear, hence Microchip proposes to generate the key of each remote from its serial number using a key-derivation function f. For customer retention, the receivers of a manufacturer should only cooperate with remotes of the same manufacturer, otherwise a competitor may sell spare remotes. Thus, f should involve a secret parameter linked to the manufacturer — the manufacturer key — that is used once in the factory to produce a new remote control, and later in the receiver when the customer registers a remote control at the secured access point. Microchip suggests several schemes, i.e. realisations of f, to derive the secret key of a remote. In the most established one, as described in [5], two KeeLoq decryptions in conjunction with the serial number are used to generate the secret key of a remote control.

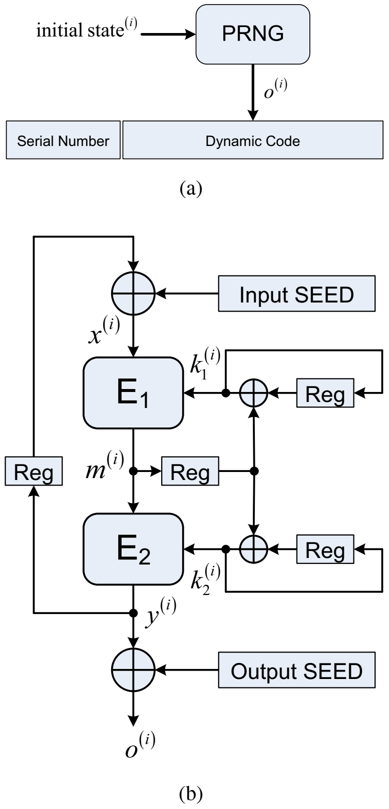
C. Vulnerabilities

After a series of crypanalytic papers on the cipher [1], [4], [7], today the KeeLoq has to be regarded as broken. The most efficient attack requires 2^{16} plain- and ciphertext pairs, however, in a KeeLoq rolling code scheme the plaintexts remain secretly in the remote control and hence a practical key-recovery does not seem possible from the mathematic point of view. In the context of side-channel cryptanalysis, this assumption does not hold any more [5], as the implementations are vulnerable to side-channel attacks and both the secret key of a remote control and the manufacturer key embedded in a receiver may get into the hands of an attacker. From the weak key derivation two main security risks arise: i) spare remote controls can be produced that are compatible with those of a certain manufacturer and ii) a remote control of this manufacturer can be cloned by eavesdropping a transmission from a distance and seconds of calculation.The evolving threat for the privacy of the customers becomes even bigger, taking the manufacturer of a rolling-code access-control into account: He himself can have a device to access any object secured with one of his products by monitoring a single rolling code sent by a genuine remote.

III. Our Proposed RKE Scheme

A. PRNG-based Dynamic Code Generator

As a remedy for the previously mentioned security risks, we propose a new RKE scheme as depicted in Fig. 2(a), whose core is a PRNG generating dynamic codes. Each time a message is to be sent by a transmitter the PRNG is clocked, thereby generating a pseudo random sequence of data. After the end of the message, it updates its internal state. The generated pseudo random sequence is - similar to a rolling-code scheme - appended with the unique serial number of the remote control before being transmitted.



(a)

(b)

Fig. 2. Structure of our proposed RKE system

Fig. 2(b) shows the basic idea of PRNG construction that is an adaptation of one presented in [17]. It contains two block ciphers, E_1 and E_2, which are connected together serially. The input of the first cipher, x, is provided by an XOR of the last output of the second cipher, y, and an "Input SEED". Moreover, the secret key of block ciphers, k_1 and k_2, are changed in each round of computation of the PRNG. That is, at the end of each round, both k_1 and k_2 are XORed with the intermediate value m, i.e., the output of E_1 and the input of E_2. Finally, the output o, which is sent to the receiver as a dynamic code, is an XOR result of output y and another seed value so-called "Output SEED" (note that we will discuss about these seeds later in Section IV). These illustrations can be summarized as follows:

$$
\begin{aligned}
x^{(i)} &= y^{(i-1)} \oplus \text{Input SEED} \\
m^{(i)} &= \mathrm{E}_1^{k_1^{(i)}}\left(x^{(i)}\right) \\
y^{(i)} &= \mathrm{E}_2^{k_2^{(i)}}\left(m^{(i)}\right) \\
o^{(i)} &= y^{(i)} \oplus \text{Output SEED} \\
k_1^{(i)} &= k_1^{(i-1)} \oplus m^{(i-1)} \\
k_2^{(i)} &= k_2^{(i-1)} \oplus m^{(i-1)}
\end{aligned}
$$

B. Receiver

The receiver construction is completely similar to the transmitter. It exactly does the same procedure as the transmitter does. Means, in contrary to the rolling code scheme, there is no decryption routine in the receiver. The receiver repeats what the transmitter has done and checks the equivalency of the received dynamic code and the computed one. Of course, the receiver must be synchronized with the transmitter regarding initial state, i.e., $m^{(i-1)}$, $k_1^{(i-1)}$, $k_2^{(i-1)}$, $y^{(i-1)}$, and SEEDs. Since the communication media in such an RKE system is typically radio frequency (or even infrared), the transmissions from the remote control to the receiver are usually noisy, and the receiver may not always see the correct code. Moreover, a button of the transmitter might be pressed far away from the receiver where it can not detect any message. Thus, the receiver must search in a space after the last valid initial state to check the validity of the received dynamic code.

More precisely, after receiving a complete message containing a serial number and a dynamic code the receiver, similarly to a rolling code scheme, checks the validity of the serial number then sets the corresponding last valid initial state to the PRNG. Later, for a certain number of rounds of the PRNG it checks the equality of the PRNG outputs and the received dynamic code. Clocking the PRNG is halted and the last valid initial state is updated once a consistency is detected. Like the rolling code scheme to prevent authenticating accidental dynamic codes, two windows are defined. First is the next 16 codes from the last valid one. Any dynamic code within this space is authenticated directly, i.e., the receiver validates the received message if an equality is detected during 16 running rounds of the PRNG. The second space contains next codes up to a certain number depending on how fast the implementation is (later in Section V we will discuss about this issue). If the received dynamic code equals one of those of this space, it will not be authenticated until two consecutive dynamic codes are received.

C. Learning

One important issue is how to register a transmitter into a receiver. Suppose that the initial values for the secret keys are stored in both the transmitter and the receiver. These two prominent secrets are proprietary values that are proportional to the manufacturer, i.e., each manufacturer has its own initial values for the secret keys. In fact, these keys play the similar role of a manufacturer key in a rolling code scheme which prevent conflict in products of different companies. A remote control must follow the below scheme, if one is going to learn it into a receiver:

1) set the last intermediate value m and output y as zero,
2) set initial values for secret keys k_1 and k_2,
3) generate random input and output seeds and send them securely for the receiver (later we will discuss about seeds in Section IV),
4) run the PRNG for one round and send the output o to the receiver[2].

In this mode, the receiver performs the same procedure,

1) receives securely input and output seeds,
2) sets last intermediate value m and output y as zero,
3) sets initial values for secret keys k_1 and k_2,
4) runs the PRNG for one round,
5) registers the new transmitter if output o and the received dynamic code are the same.

IV. Security Analysis

A. Cipher Analysis

Since the construction of our RKE scheme is based on the PRNG presented in [17] and the adversary can only observe an XOR result of the PRNG output y and a seed, the theoretical analyses given in [17] to evaluate security of the PRNG are true for our proposed RKE system too. In short, supposing that block ciphers E_1 and E_2 are ideal ciphers, the PRNG and hence the RKE is theoretically secure. In other words, no reasonable adversary can distinguish the uniform distribution on internal state (dynamic secret keys) from the visible uniform distribution on the output (either y or o).

B. Physical Security Analysis

As shown in [17] side-channel key recovery attacks, which require a set of query measurements under an assumption that the secret key is fixed, can not reveal the dynamic secret keys k_1 and k_2. Indeed, the re-keying strategy, or in general using each session key for once, prevents such side-channel attacks, e.g., DPA. However, profiling attacks, that are capable of revealing the secrets using a single measurement, may challenge the PRNG and recover the secret keys of a round leading to completely break the system. One solution presented in [17] is to implement the block ciphers on 64- or 128-bit architecture, thereby making the profiling as hard as possible, and increasing the switching noise for smaller templates, e.g., 8- or 16-bit. Of course this idea is appropriate for FPGA and ASIC implementations, and is not applicable when the implementation platform is a microcontroller, e.g., commercial RKE systems.

One important difference between the construction of our RKE scheme and the aforementioned PRNG is due to the seeds. In the PRNG the seed playing the IV role is public, and the output of the PRNG is sent to the other party. However, in our RKE scheme input seed must be kept secret, and the output of the sender is an XOR result of the PRNG output and a confidential output seed. As mentioned, if the seeds are public, a side-channel adversary can reach the secret keys by profiling the target device. In the following we discuss how the secrecy of the seeds can protect the implementation in the presence of template attacks.

Since the output seed is secret, what a profiling side-channel adversary using the leakage of the block cipher E_2 can achieve is a dynamic secret key k_2 mixed by the output seed. Usually block ciphers, e.g., Rijndael, use a whitening scheme where final output is XORed with a rounkey at the last stage. Then, a template attack targeting the last (N-th) roundkey, $Rkey_2^N$, using the leakage of the last substitution box, $S(Rkey_2^N \oplus y)$,

[2] Since in this scenario, the transmitter changes its own initial state, it should be designed in order to prevent customers from accidentally putting a remote control into this mode. Otherwise, it should be learned again into the target access point.

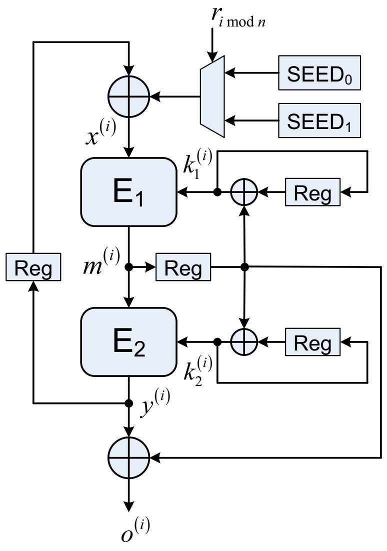


Fig. 3. Final structure of our proposed RKE system

(or a non-linear function) recovers the XOR result of the roundkey and the output seed, $Rkey_2^N \oplus \text{Output SEED}$. However, a template-based DPA attack can reveal the output seed by considering the leakage of the XOR between the PRNG output y and the output seed. Since template-based DPA attacks need a set of measurements with the same secret, the output seed must be dynamic to prevent such an attack.

The reason to use the input seed in our RKE scheme is to prevent overlapping the stream generated by different remote controls. On the other hand, it should not be a function of the serial number, otherwise like what presented for KEELOQ key derivation schemes in [5] it might be revealed by a side-channel key recovery attack. As illustrated in Section III-C, the remote control must generate a random input seed and sends securely to the receiver. Otherwise, it would be public and lets an adversary challenge the PRNG by a template attack or even a template-based DPA attack (again supposing that the implementation is an 8-bit microcontroller). Fig. 3, which is an adaptation of "Secure Initialization" of the PRNG presented in [17], shows the final structure of our proposed RKE scheme. When a transmitter is at the learning mode, generates a n-bit random number r, runs the PRNG for n rounds without having any output o. Then, it sends r and the last output o for the receiver. While r is public, both SEED0 and SEED1 are still secret and could be selected by the manufacturer. In contrary to the secure initialization of the PRNG, r is used in every round of execution of the RKE, i.e., each time (round i) a message must be sent by the remote control, ($i \bmod n$)-th bit of r is contributed in the computation.

Further, the intermediate value m is used as "Output SEED". Since it is changed in every execution round of the PRNG, a template-based DPA could not collect more than one query with the same output seed, and hence would be ineffective.

The final open problem is due to the fact that a template attack may challenge the system using the leakage of the last XOR operation on the PRNG output y and the output seed m on an 8-bit architecture. Supposing a HW leakage model for the target implementation, an adversary which previously learned $\mathrm{HW}(y)$ (or $\mathrm{HW}(m)$) as an input of the aforementioned XOR operation for all y (or m) values is going to get some information about y (or m) knowing a specific $\mathrm{HW}(y^{(i)})$ (or $\mathrm{HW}(m^{(i-1)})$) and $o^{(i)} = y^{(i)} \oplus m^{(i-1)}$. As illustrated in [17], supposing a noise free situation the success rate would be $(b+1)/(2^b)$ in a b-bit architecture, i.e., in our case $b = 8$ and success rate turns into 0.035. However, the adversary can use an identity leakage function instead of the HW model. According to our practical experiments on an 8-bit Microchip PIC microcontroller, when the target leakage is due to a linear function, e.g., an XOR operation, the success rate is about 0.20 using the identity leakage function, i.e., profiling $o = y \oplus m$ for all possible values of y and m[3]. Interestingly, the success rate of recovering y (or m) and hence k_2 and etc. with a length of l bits is $(0.035)^{l/8}$ and $(0.20)^{l/8}$ for respectively HW and identity leakage models. Supposing a typical 128-bit length for the key and block size of the block ciphers, these values would be turned into 2^{-77} and 2^{-37} respectively.

Note that since the construction of the transmitter and the receiver are the same, their physical security against power-analysis attacks either those need a set of queries or profiling ones are the same. Means, if a side-channel adversary can not challenge the transmitter, will not be able to reveal any secret from the receiver too.

As pointed out, the manufacturer of a KEELOQ rolling code system is capable of contravening the customers' privacy by monitoring a transmission of a genuine remote control. However, in our proposed RKE system knowing the initial value of the secret keys k_1 and k_2 does not help the manufacturer to follow the streams sent by the transmitter till the random initialization value r is secret. The only possibility to know r is to monitor its transmission during the learning mode.

V. IMPLEMENTATION

In order to have an insight about implementation of our RKE scheme and also to have an example platform for practical analysis, we have built a complete system based on the illustrated scheme. Since the implementation platform of commercial RKE systems is usually a microcontroller and AES Rijndael is an efficient block cipher for such 8-bit architectures, we have used AES-128 as block ciphers E_1 and E_2. Consequently, input and output seeds, intermediate value m, and the dynamic code o would be 128-bit values. Further, we have selected r as a 128-bit value used for generating a random input seed.

As mentioned before, running speed of the implementation of block ciphers is of highly importance in the receiver. Since we have used one of the Microchip PIC18F family microcontrollers, e.g., [15], in both the transmitter and the receiver, we needed a high speed implementation of the AES-128 cipher. To

[3] Note that the success rate in this case depends strongly on the noise of the measurement setup as well as on how good profiling phase has been done.

our knowledge, one of the fastest implementations on the same platform can be found in [19] which needs 3137 execution cycles to run an AES-128 encryption routine including computation of roundkeys on-the-fly. Means, using a typical 20MHz crystal as the main resonator the aforementioned algorithm is executed in about 627μs. Since two instances of the encryption must be executed in each round of the PRNG, we could run it at least 512 rounds in a second, of course including the computation time for adding seed values and updating initial state. As a result, in our implementation we could adjust the length of the second window, described in Section III-B, as 512 codes. In other words, if the button of a remote control is pressed roughly 500 times far away from the receiver, still it can be authenticated in a second by sending two consecutive messages.

Another important issue in implementation of RKE systems is due to the power consumption. Since the remote controls are usually battery operated, it is of crucial importance to have a low power design for the transmitter. Two AES encryptions are the biggest functions that are executed in a transmitter for sending each dynamic code. In average generating each dynamic code in our implementation needs $10\mathrm{mA}$ for $2\mathrm{ms}$ and the radio frequency communication part needs $50\mathrm{mA}$ for $450\mathrm{ms}$. Using a typical $3\mathrm{V}/600\mathrm{mAh}$ battery a transmitter can operate for more than $90\,000$ times. On the other hand, since the receiver is usually installed in an access point, it is supplied by electricity and is not challenged by running the PRNG for 512 times by receiving an invalid message.

VI. Conclusions

In this article we have presented an application for the secure PRNG proposed in [17]. In addition to the conventional DPA-resistance techniques which are applied at the implementation level, the PRNG has been designed to be secure against side-channel attack independently of the method of implementation. The RKE system which has been presented in this paper is an adaptation of the secure PRNG and is of highly importance in the presence of recent vulnerabilities of a widely used RKE system namely KeeLoq rolling code scheme [5]. Our proposed RKE scheme not only is inherently secure against side-channel key recovery attacks, but also satisfies the privacy of the customers, i.e., in contrary to the KeeLoq rolling code scheme a manufacturer of our RKE system is not capable of accessing the secured objects developed by its own products.

Since RKE systems are usually constructed using general-purpose 8-bit microcontrollers, employing purely the PRNG leads to vulnerability to template attacks. Thus, we have slightly modified its structure and the assumptions on seed values to make a resistant implementation against side-channel attacks including profiling ones even on an 8-bit architecture.

This work would be an example to show there are alternative choices for conventional countermeasure, e.g., masking, hiding, and those applied at layout level, that are not taken into consideration so much by the manufacturers because of their time, area, and power overheads. Further, designing new systems to inherently counteract side-channel attacks devoid of a particular implementation is of highly interest from cryptographers' as well as industrial crypto engineers' point of view.

References

[1] A. Bogdanov. Attacks on the KeeLoq Block Cipher and Authentication Systems. In *RFIDSec 2007*. http://rfidsec07.etsit.uma.es/slides/papers/paper-22.pdf.

[2] S. Chari, J. R. Rao, and P. Rohatgi. Template Attacks. In *CHES 2002*, volume 2523 of *LNCS*, pages 13–28. Springer, 2002.

[3] C. Clavier, J.-S. Coron, and N. Dabbous. Differential Power Analysis in the Presence of Hardware Countermeasures. In *CHES 2000*, volume 1965 of *LNCS*, pages 252–263. Springer, 2000.

[4] N. T. Courtois, G. V. Bard, and D. Wagner. Algebraic and Slide Attacks on KeeLoq. In *FSE 2008*, volume 5086 of *LNCS*, pages 97–115. Springer, 2008.

[5] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, and M. T. M. Shalmani. On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoq Code Hopping Scheme. In *CRYPTO 2008*, volume 5157 of *LNCS*, pages 203–220. Springer, 2008.

[6] C. Herbst, E. Oswald, and S. Mangard. An AES Smart Card Implementation Resistant to Power Analysis Attacks. In *ACNS 2006*, volume 3989 of *LNCS*, pages 239–252. Springer, 2006.

[7] S. Indesteege, N. Keller, O. Dunkelman, E. Biham, and B. Preneel. A Practical Attack on KeeLoq. In *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 1–18. Springer, 2008.

[8] M. Kasper, T. Kasper, A. Moradi, and C. Paar. Breaking KeeLoq in a Flash. In *AFRICACRYPT 2009*, volume 5580 of *LNCS*, pages 402–419. Springer, 2009.

[9] LinearCorp. MegaCode and MultiCode Systems. http://www.linearcorp.com/radio_megacode.html.

[10] S. Mangard, E. Oswald, and T. Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, 2007.

[11] S. Mangard, N. Pramstaller, and E. Oswald. Successfully Attacking Masked AES Hardware Implementations. In *CHES 2005*, volume 3659 of *LNCS*, pages 157–171. Springer, 2005.

[12] Microchip. HCS301 KeeLoq Code Hopping Encoder Data sheet. http://ww1.microchip.com/downloads/en/DeviceDoc/21143b.pdf.

[13] Microchip. HCS410, KeeLoq Code Hopping Encoder and Transponder Data Sheet. http://ww1.microchip.com/downloads/en/DeviceDoc/40158e.pdf.

[14] Microchip. PIC16C5X Data Sheet. http://ww1.microchip.com/downloads/en/DeviceDoc/30453d.pdf.

[15] Microchip. PIC18FXX2 Data Sheet. http://ww1.microchip.com/downloads/en/DeviceDoc/39564c.pdf.

[16] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen. A Side-Channel Analysis Resistant Description of the AES S-Box. In *FSE 2005*, volume 3557 of *LNCS*, pages 413–423. Springer, 2005.

[17] C. Petit, F.-X. Standaert, O. Pereira, T. G. Malkin, and M. Yung. A Block Cipher based Pseudo Random Number Generator Secure Against Side-Channel Key Recovery. In *ASIACCS 2008*, pages 56–65. ACM, 2008.

[18] T. Popp and S. Mangard. Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints. In *CHES 2005*, volume 3659 of *LNCS*, pages 172–186. Springer, 2005.

[19] The Hardware Side of Cryptography. Fast AES Implementation on PIC18F4550. http://edipermadi.wordpress.com/2008/06/17/fast-aes-implementation-on-pic18f4550.

[20] K. Tiri, M. Akmal, and I. Verbauwhede. A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards. In *ESSCIRC 2002*, pages 403–406, 2002.

[21] J. Waddle and D. Wagner. Towards Efficient Second-Order Power Analysis. In *CHES 2004*, volume 3156 of *LNCS*, pages 1–15. Springer, 2004.

[22] S. Yang, W. Wolf, N. Vijaykrishnan, D. N. Serpanos, and Y. Xie. Power Attack Resistant Cryptosystem Design: A Dynamic Voltage and Frequency Switching Approach. In *DATE 2005*, pages 64–69. IEEE Computer Society, 2005.