Security Analysis of a Widely Deployed Locking System

Michael Weiner
Technische Universität München
m.weiner@tum.de

Maurice Massar
Technische Universität Kaiserslautern
massar@unix-ag.uni-kl.de

Erik Tews
Technische Universität Darmstadt
erik@datenzone.de

Dennis Giese
Technische Universität Darmstadt
dennis.giese@stud.tu-darmstadt.de

Wolfgang Wieser
Ludwig-Maximilians-Universität München
wolfgang.wieser@physik.uni-muenchen.de

ABSTRACT

Electronic locking systems are rather new products in the physical access control market. In contrast to mechanical locking systems, they provide several convenient features such as more flexible access rights management, the possibility to revoke physical keys and the claim that electronic keys cannot be cloned as easily as their mechanical counterparts. While for some electronic locks, mechanical flaws have been found [1], only a few publications analyzed the cryptographic security of electronic locking systems [2, 3]. In this paper, we analyzed the electronic security of an electronic locking system which is still widely deployed in the field.

We reverse-engineered the radio protocol and cryptographic primitives used in the system. While we consider the system concepts to be well-designed, we discovered some implementation flaws that allow the extraction of a system-wide master secret with a brute force attack or by performing a Differential Power Analysis attack [4] to any electronic key. In addition, we discovered a weakness in the Random Number Generator that allows opening a door without breaking cryptography under certain circumstances. We suggest administrative and technical countermeasures against all proposed attacks.

Finally, we give an examination of electronic lock security standards and recommend changes to one widely used standard that can help to improve the security of newly developed products.

CCS'13, November 4–8, 2013, Berlin, Germany.

ACM 978-1-4503-2477-9/13/11 ...$15.00.
http://dx.doi.org/10.1145/2508859.2516733.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection (e.g., firewalls)*; C.2.1 [**Network Architecture and Design**]: Wireless communication

Keywords

Cryptography; DES; Embedded Security; Physical Security; Side-Channel Attack; Power Analysis; PRNG; Locking System.

1. INTRODUCTION

Locks might be the most frequently used symbol icon when it comes to explaining IT security and cryptography. While mechanical locks have evolved for more than two millenniums, digital access control systems have only been increasing their market share in the last decades. Such systems allow greater flexibility when it comes to managing access control for a large number of users for a facility and easy revocation of lost keys. They also provide the user with the advantage that only one single token is needed for accessing many different locks.

Furthermore, electronic locking system manufacturers claim that it is difficult to copy or clone keys, or – in other words – forge the authenticity of key owners. In general, the authenticity of a person or object can be ensured by the means of knowledge, ownership or inherent properties [5]. In the case of almost all mechanical locks, the ownership of a key can be tracked down to the knowledge of its shape that can be observed and cloned easily. Some attacks even focus on reconstructing mechanical keys from newspaper or television images [6]. This is not possible for electronic access tokens such as smartcards or electronic transponders: even though they are supposed to contain secret keys, cryptographic algorithms and protocols shall ensure their secrecy.

However, the exact functionality of electronic locks is often undocumented in contrast to mechanical locks with public technical principles. This violates Kerckhoff's principle [7] and makes it difficult for third parties to independently evaluate the security of electronic locking systems. In the past,

several widely deployed electronic security systems with a lack of public documentation were found out to be vulnerable against mathematical or side-channel attacks [2, 8, 9].

While NXP (Mifare Classic) and Legic (Prime) have improved the security of their new products after a public analysis of their products [8, 9], we could not find much information about SimonsVoss, one of the leading companies for for electronic locking systems in Europe. Their products include electronic locks, the corresponding access tokens, as well as equipment that is used to run and integrate the system into the existing infrastructure of a company. Reference customers include banks, hospitals, transportation companies, power plants, food and water supply companies, courts, prisons and military installations.

Currently, there are two product lines from SimonsVoss on the market. *System 3060 Generation 1 (G1)* has been available since 1997 [10] and is discouraged for new installations and replaced by their new main product line, *System 3060 Generation 2 (G2)* introduced in 2007 [11]. However, G1 is still sold for existing G1 setups. G1 and G2 differ in the communication protocol and the use of cryptographic primitives, but G2 locks and transponders have the same outer appearance as their G1 counterparts sold as of 2007 [10].

The cryptographic security of Generation 2 has recently been analyzed by Strobel et al. [3]. The researchers reverse-engineered the cryptographic primitives and found out that the random number generator in their locks allowed the derivation of transponder secrets, which can be used to create a valid transponder without having physical access to it. According to the manufacturer, this vulnerability no longer exists in newly deployed locks and a patch is available for existing installations.

In this paper, we present our analysis of the SimonsVoss System 3060 *Generation 1 (G1)* electronic access control system. In Section 2, we outline the general structure of the system and the wireless protocol. Section 3 describes a low-cost method to reverse engineer the system and the used cryptographic methods. A discussion of possible attack vectors can be found in Section 4. We outline multiple realistic attack scenarios that all allow an attacker to open a lock without the consent of the operator with a justifiable effort. Of course we also present countermeasures against the attack in Section 5. In Section 6, we recap our attack scenarios and match them with recognized standards for electronic locks.

2. SYSTEM CONCEPT

This section gives an overview of the SimonsVoss System 3060 G1 by describing the devices involved as well as their identifiers. Furthermore, the communication between the devices is outlined and the important secrets are explained.

2.1 Devices

The analyzed locking system consists of at least three types of devices.

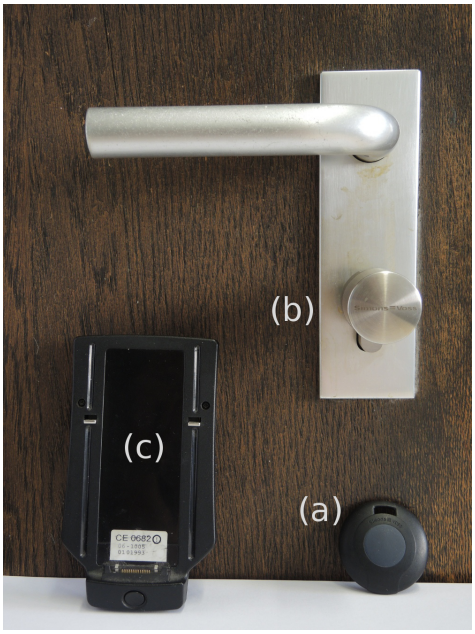

Figure 1: Transponder (a), Lock (b) and Configuration Device (c)

2.1.1 Transponder (TR)

The physical keys of the system are called transponders. They have one button that initiates a data transfer to a lock or configuration device when pushed. Transponders can be programmed for up to three locking systems. They do not have globally unique serial numbers that would be visible during regular operation. Transponders will be abbreviated TR.

2.1.2 Lock (LK)

Mechanical locks as well as electrical relays are unlocked or activated by an authorized transponder. In the following, cylinders, relays and other devices against which a transponder authenticates itself will be called lock (LK).

2.1.3 Configuration Device (CD)

Configuration devices (CD) are used to program locks and transponders.

Figure 1 shows a transponder and a PalmCD configuration device next to a mounted lock. The PalmCD is discontinued and has been replaced by a product named SmartCD, but it is compatible with G1 installations.

In addition to those three types of devices, additional components like a central user account database and communication links to emergency and alarm systems might be added to the system.

2.2 Identifiers

2.2.1 Overview

The Transponder ID (`TID`) is a 13 bit number identifying transponders within a locking system. Its scope is constrained to the locking system. Locks are identified by their 14 bit Lock ID (`LID`), which only needs to be unique within the locking system. Finally, the Locking System ID (`SID`) identifies a locking system. Its length is not specified, but

we assume it to be 14 bits. Even though the SID can be freely configured in some cases, the ID should be globally unique. During our radio protocol analyses, we figured out that all of these IDs are 16 bit values on the radio layer. The additional bits are used as parity values and flag bits we did not analyze any further.

2.2.2 Special TIDs

There exist special TIDs that are used for configuration and other purposes. One of these TIDs serves the purpose of opening all locks of one locking system in emergency situations, i.e. *all* locks consider this TID as authorized independently of the access control configuration. While the original configuration software does not allow programming transponders with this ID, we found out that a custom program communicating with an original Configuration Device can program this "emergency opening" TID, or, in other words, create a master transponder authorized for all locks of a particular locking system.

2.3 Communication

2.3.1 Purpose

From an abstract point of view, communication between TR, LK and CD serves three main purposes.

- Programming LK or reading out log files
- Programming TR
- Authenticating TR against LK

In this paper, we will focus on the last type of communication as the attacks are only based on the last mentioned type.

2.3.2 Modulation and Encoding

The components communicate over a 25 kHz radio signal. The bits are encoded using Differential Manchester Encoding, the encoded symbols are modulated using a simple On-Off-Keying scheme.

The low carrier frequency of the signal allows recording with an ordinary laptop computer with a sound card and a coil connected to the microphone input of the card. Also, this equipment can be used to transmit custom commands with a coil connected to the speaker output. Please note that there exist sound cards using hardware low-pass filters. Such cards cannot be used for our analyses.

2.3.3 TR authentication

Unlocking a cylinder implies running a challenge-response protocol with a 32 bit challenge C and a 40 bit response R in which the transponder authenticates itself to the lock. Figure 2 depicts an authentication of TR against LK. One arrow represents one message consisting of preamble, payload and an 8 bit odd parity value.

The status flags of the transponder include a timezone that can be used to grant time-dependent access rights. We did not analyze the lock status flags any further.

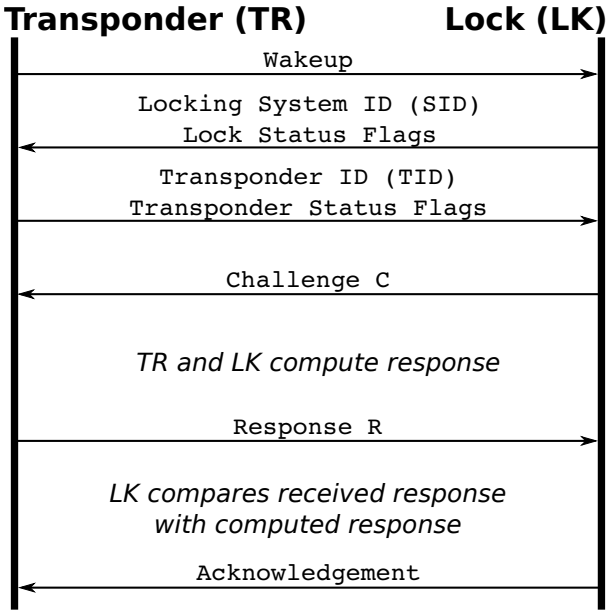

Figure 2: Transponder Authentication Protocol

2.4 Secrets

2.4.1 System Secret

All components of one locking system share one common locking system key [12]. Anyone knowing this key can, for example,

- program new transponders with the correct system key and a TID of choice. These transponders can be used to impersonate every user in the system.
- re-configure locks, add new authorizations or remove existing authorizations and extract logfiles.
- program a master key for the system that will open any lock regardless of the current configuration of that lock.

2.4.2 Superordinate Keys

There also exist superordinate keys that aim at increasing the convenience of large-scale configurations. The system concept allows special transponders that are authorized for more than three locking systems that a transponder can regularly manage. Such transponders could be used, for example, by firefighers or cleaning staff. Conceptually, this is implemented by reserving three different TIDs to each of which one distinct "superordinate key" is assigned. When these TIDs, which are referred to as "red", "green" and "blue" in the documentation [12], authenticate against a lock, the corresponding superordinate key serves as a replacement for the system secret during the response calculation. Superordinate keys cannot be used for programming locks.

3. REVERSE-ENGINEERING

During the radio protocol analysis, we could not see any kind of simple pattern in the response messages. To successfully attack this challenge response authentication scheme, we had to reverse engineer the cryptographic algorithms that were used to generate the responses to the challenges.

By a prior observation of different challenge-response pairs with different TIDs and SIDs we knew that the TID (16 bit), challenge C (32 bit) and a 56 bit key k were used as an input to the challenge computation, while the SID was not.

We assumed that the scheme is based on DES [13] for two reasons:

- Configuring different locking systems with identical configuration, but different passwords, leads to 64 differing bits in the communication between PC software and CD, excluding additional checksums. Out of those 64 bits, every eighth bit could be identified as a parity bit. This pattern is used in DES to reduce the effective key length from 64 to 56 bits. This key is a global system secret and used for all locks and transponders in the system.
- A Simple Power Analysis (SPA) [4] revealed 16 repeating sequences in an averaged power trace. The assumption seems reasonable that each sequence corresponds to one of the 16 DES rounds. A trace of the response computation is depicted in Figure 3. The magnetic field intensity was measured close to the power pin of the microcontroller of the transponder to serve as an indicator of its current consumption.

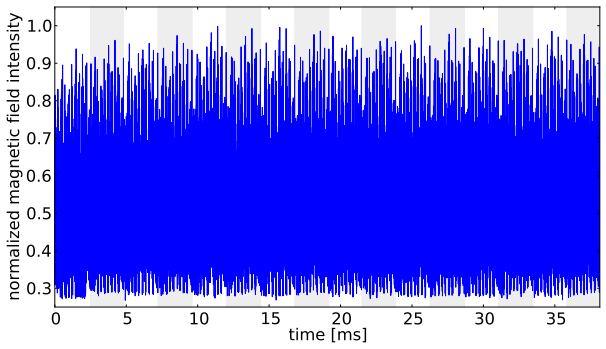

Figure 3: Trace of the supposed DES computation

However, an unknown function f_{in} is used at the input to prepare a 64 bit plaintext using TID and challenge C, and another unknown function f_{out} extracts the 40 bit response R from the 64 bit ciphertext:

$$R = f_{out}(DES(f_{in}(TID, C), k))$$

To reverse engineer these algorithms we had a number of choices: Previous successful attacks [14, 15] used invasive methods against the chip that involve opening the package. The program code can then be extracted by microprobing the ROM data bus. Alternatively, one may read out the ROM directly using microprobing or clear the memory protection bits using UV light or a laser/focused ion beam.

As those approaches are rather difficult to execute, we decided to start with an easier approach: a blind power glitching attack. Using glitching, one may skip one or more instructions executed on the chip or cause an incorrect or partial execution of these instructions. The output of such an incorrect execution of the program flow might reveal details of the algorithm executed on the chip. In contrast to many other power glitching attacks that use an FPGA, all we required for a successful attack was an AVR microcontroller generating the glitches and controlling the I/O pins of the PIC microcontroller under attack. For this purpose, we de-soldered the microcontroller from a transponder and placed it into a custom setup as shown in Figure 4.

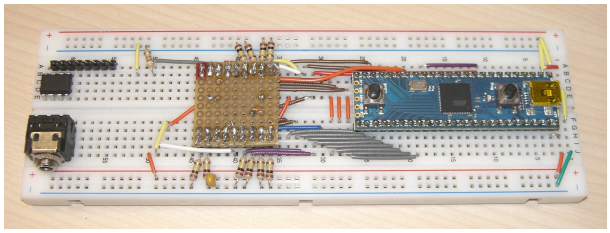

Figure 4: Power glitching setup

This attack was aimed at revealing the unknown functions f_{in} and f_{out}, verifying if DES is really used during the response computation, as well as finding possible modifications of the DES standard.

We assumed that a conditional branch instruction is used to jump from the end of one supposed DES round back to its beginning unless 16 rounds have already been executed. Our intention was locating this instruction and causing the power glitch to skip this instruction, such that only one round is executed instead of 16. As we did not know the exact position of the conditional branch instruction to attack, we iterated over all clock cycles during the response computation and observed the execution time until the controller started to transmit the response. We found out that when glitching a certain clock cycle, the controller began transmitting the response after roughly 1/16 of the expected number of clock cycles.

As the next step, we needed to reconstruct the dependency between input and output bits. For this reason, we observed the glitched response r of a randomly chosen challenge C, as well as the response r' of a challenge $C' = C \oplus 2^i$ where one bit of C was inverted. We repeated this procedure for all bit positions i and for a large number of challenges C. This procedure was repeated for all other possible inputs to the response computation, such as the SID, TID and the timezone.

The corresponding response pairs r and r' differed in 1, 4, 5, 8 or 9 bits, depending on the position i of the flipped input bit. A deeper analysis revealed that this corresponds to a single round of DES. A single bit difference corresponds to the left side of the Feistel network, and a 4 to 9 bit difference corresponds to the right side of the Feistel network where the bit was used as input to one or two S-Boxes.

Analyzing these outputs, we were able to derive the unknown mappings f_{in} and f_{out} that generate a DES plaintext and extract the response R. We have decided not to publish these mappings because they are not required to understand the rest of this paper, but would make it easier to reproduce our results to actually cause damage by attacking a G1 locking system for non-academic purposes.

4. ATTACKS

After having reverse engineered the authentication scheme, we were able to discover multiple practical attacks against the system.

4.1 PRNG

After we had implemented our first radio sniffer and gained a basic understanding of the radio protocol, we were able to record the challenges sent by the lock to the transponder.

4.1.1 Introduction

Subsequent challenges we observed from regular authentication runs appeared to be subsequent states of a 32 bit Linear Feedback Shift Register (LFSR) being clocked twice between two challenges; only the seven least significant bits of a challenge could not be expressed as a linear function of the previous challenge.

To analyze the behavior of the PRNG more deeply, we implemented a transponder emulator that continuously requested challenges but aborted each authentication run after receiving the challenge. In contrast to our first observations, challenges obtained in this way seemed to be states of an LFSR that only is clocked once – not twice – between each challenge. Furthermore, we observed that now, *all* bits could be expressed as a linear function of the previous challenge. The following list of subsequent challenges in binary representation shows the simplicity of the scheme.

```
00000111100100001101101100110011
10000011110010000110110110011001
01000001111001000011011011001100
10100000111100100001101101100110
01010000011110010000110110110011
10101000001111001000011011011001
01010100000111100100001101101100
10101010000011110010000110110110
11010101000001111001000011011011
11101010100000111100100001101101
```

The next challenge sent by the lock was just the previous challenge with all bits rotated by one, and one bit updated with a new value. This single bit is generated by an LFSR in Fibonacci configuration with the characteristic polynomial $x^{32}+x^{31}+x^{29}+x^{28}+x+1$. A block diagram of this LFSR is shown in Figure 5.

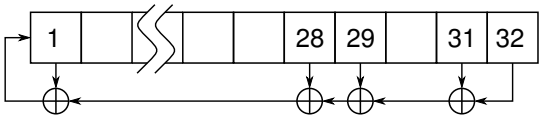

Figure 5: Block diagram of LFSR

Surprisingly, the feedback polynomial is not irreducible, and can be written as $(x^{31}+x^{28}+1)(x+1)$. We observed that it generates two non-trivial cycles of length $2^{31}-1$.

Regular updates of the LFSR can be modeled as XORing its seven least significant bits with a random value and then clocking it twice. However, 1205 non-linear updates we analyzed never left the initial LFSR cycle, so we assume that at latest after $2^{31}-1$ challenges, a challenge is repeated in all cases.

A flow chart depicting our simplified model of the behavior of locks during transponder authentication is shown in the Appendix in Figure 9.

4.1.2 Attacks on the PRNG

The weak PRNG can be used to attack the lock in various scenarios. We call the first attack a *return then unlock* attack. Imagine a system where people are allowed to borrow transponders for a limited time. For example, housekeeping might be allowed to borrow a transponder during the day to clean up an office. Afterwards the transponder is returned.

Due to the PRNG design, an attacker can predict the next challenge after having executed an aborted protocol run with the lock. The attacker, who is in temporary possession of a transponder, can use this transponder to generate a valid response for the predicted challenge. An attacker may borrow such a transponder, execute an aborted protocol run with a door he would like to unlock later, and then return the transponder. Later, he returns to the door and opens it with the previously generated response. Having already returned the transponder for this door gives him plausible deniability. Instead, the person who was in possession of the transponder when the door was unlocked will be suspected of having opened the door.

If the lock was opened once, the attacker cannot predict seven of the 32 challenge bits, if the lock was opened twice, he is missing nine bits, and so on. In the general case, the attacker needs to collect 2^{2n+5} challenges to be prepared for n regular openings ($1 \leq n \leq 13$) before he wants to open the door using his predicted challenges and the appropriate responses.

The attack can be extended to a more general attack. If a protocol run has been interrupted, the PRNG state is simply updated by clocking the LFSR. This means that all possible states of the LFSR cycle have an order that is determined by the feedback polynomial. Assume that an adversary has control over a transponder for a longer time such as, for example, a weekend. He can query the transponder with 2^k challenges, which all have the same distance in the LFSR state order (2^{31-k}). In average, we were able to get a response every 1.5 seconds, so that 2^{17} challenge-response pairs can be gathered in less than 55 hours. The attacker can now return the transponder.

If the attacker would like to open a specific lock, he runs one protocol run with the door to obtain the current PRNG state. He will be less than 2^{14} challenges in LFSR order away from a challenge he already knows the response for. He now needs to execute 2^{13} linear updates of the PRNG of the lock in average, until the lock chooses a challenge he knows the response for and can open the lock. If a single update takes him 3 seconds, he can open the door in less than 7 hours in average.

In general, the creation of a lookup-table of 2^k entries will take $1.5 \cdot 2^k$ seconds assuming that capturing one challenge-response pair requires roughly 1.5 seconds. The average time to open the lock is $1.5 \cdot 2^{31-k-1}$ seconds. While this attack is quite simple and does not require many computing resources, the consumed time is still considerable.

4.2 Brute-Force

One of the simplest approaches to attack the system is a brute-force search for the correct key. As only 40 bits of the DES output are used for the response, the key cannot be determined uniquely using a single challenge-response pair, so at least two pairs are required. An adversary can use the first challenge-response pair to determine all DES keys that generate that response from the challenge, giving the adver-

sary 2^{16} key candidates in average. These candidates are then tested against a second challenge-response pair. With a probability close to 1, only one of the 2^{16} keys will also generate the correct response for the second challenge.

The effort for the first state is roughly equivalent to a full DES brute force attack, while the second stage, i.e. testing 2^{16} keys, only requires negligible resources. Doing a full DES brute force attack is affordable for almost any adversary today. For example, CloudCracker [16] runs an FPGA cluster that can search the entire DES key space in 24 hours. This service is currently offered for a price of 100 US$.

While the cost of such an attack is quite low, it still requires either involving a third party, or buying expensive equipment to perform the attack on site.

4.3 Differential Power Analysis

Differential Power Analysis (DPA) [4] is an attack aimed at revealing the secret key of cryptographic operations that targets the hardware on which a cryptographic algorithm is running. For attacking a device, an adversary needs to perform multiple cryptographic operations using the same key, but different input data. For each operation, he records a *power trace* containing the amplitude of a *leakage source* over time. Assuming that different data values processed by a device lead to a different power consumption, the adversary can chose the current consumption of a device as the leakage source; another possible leakage source is the electro-magnetic radiation [17].

The adversary can then create hypotheses about *intermediate values* of the cryptographic operation that depend on a known value (usually the plain- or ciphertext of an encryption run) and a small portion of the unknown secret key. These hypotheses can be tested using mathematical methods that combine them with the power traces. The best fitting hypothesis eventually reveals a small portion of the key. This process can be repeated for all intermediates values that depend on varying input data such that in the best case, the full key can be recovered.

When a transponder authenticates against a lock, both lock and transponder perform the modified DES algorithm as described in Section 3 with the same key and plaintext. While it is known that unprotected devices performing DES encryption can be attacked using a DPA attack, our scenario does not allow a complete standard attack because some bits of the plaintext are constant and cannot be altered. In this section, we have adapted to those restrictions and show that a DPA in combination with an exhaustive search of the remaining key space can be applied to transponders in a very short time.

From a technical perspective, it is advisable to perform a DPA attack on locks: The adversary can freely chose a different `TID` for each trace. Attacking a transponder implies that the `TID` is constant for all traces, such that out of the 64 bit plaintext, 32 bits remain constant while only the 32 bit challenge generates differences.

Attacking transponders is more attractive to an adversary from a practical perspective, as they can usually be carried away and placed into a measurement setup without attracting any attention. Therefore, we chose to implement a DPA attack on a regular transponder programmed with a `TID` and the 56 bit system key.

We designed our measurement setup to allow real-world attacks. The following design rules allow attacking programmed transponders and avoid the necessity of physical modifications to it. Using those rules, it is possible to create a measurement setup of which the application to a specific transponder cannot be detected by regular visual inspection of the transponder circuit board (e.g. there are no ripped up wires on the transponder PCB).

- We used the electro-magnetic radiation as a leakage source [17].
- We emulated pushing the transponder button by the use of available test pads on the circuit board.
- We used the one of the TX outputs of the microcontroller as a trigger signal for the oscilloscope to time-synchronize traces.
- While in most DPA setups, the device under test is provided with an external clock that is synchronized to the sampling clock of the oscilloscope, we captured the transponder clock for each trace and achieved synchronization by the means of post-processing in software.

We conducted a proof-of-concept implementation of the measurement setup. An electro-magnetic probe at the power supply pins of the MCU was used to capture the radiation. A sophisticated adversary can use a bed-of-nails, which has been known from PCB production tests for decades [18], for the required electrical connections to minimize the trace he leaves; our proof-of-concept implementation is satisfied by using soldered wires. The measurement setup is depicted in Figure 6.

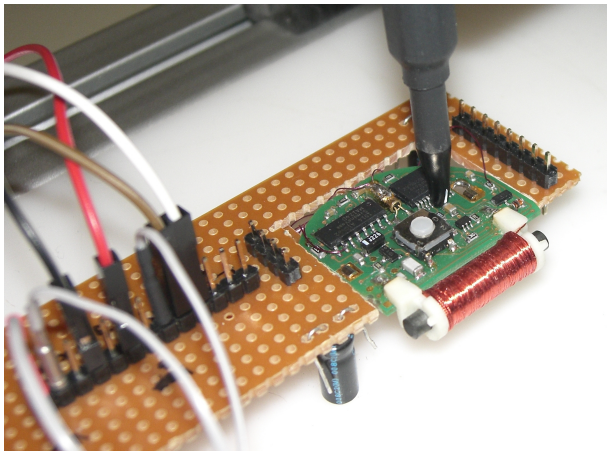

Figure 6: Proof-of-concept DPA setup of transponder

We decided to start with a standard Correlation Power Analysis (CPA) [19] on DES using the S-Box outputs of the first round as intermediate values. As a power model, we used the Hamming Weight of the intermediates.

Using a transponder as a target device leads to the fact that 32 of the 64 bits are constant, and only the 32 bit challenge provides variation of the inputs. Two of the eight DES S-Boxes are completely fed by the challenge. We refer to them as S_A and S_B. For two other S-Boxes that we name S_C and S_D, five of the six input bits are fed by the challenge, while one bit is constant.

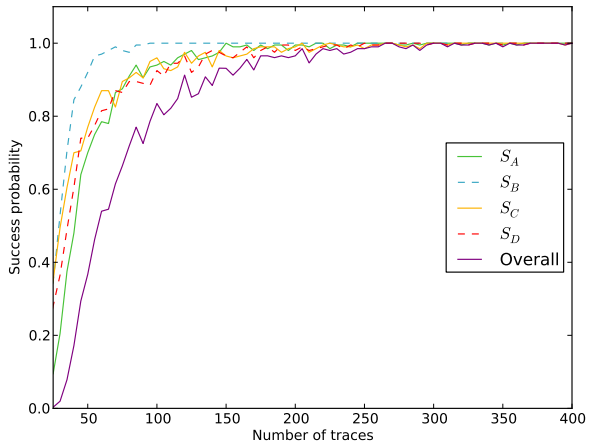

Figure 7: Success probability over number of traces

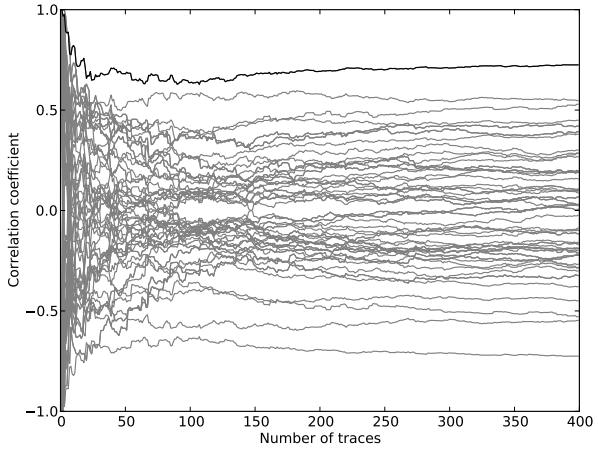

Figure 8: Correlation of S_D subkey candidates over traces

Figure 7 shows the success probability over the number of traces. It was created using 3367 traces in total. The success probability was determined by observing the relative occurrence of cases in which the correct subkey candidate position is above a certain threshold, as described below. Each point on each of the four S-Box curves was created by randomly selecting the appropriate number of traces and observing the relative occurrence over 200 runs.

We found out that the subkeys feeding S_A and S_B can be found out after roughly 100 to 150 traces. For those S-Boxes, the success probability is defined as the probability that the correct subkey has the highest absolute value of correlation coefficient for all possible subkeys. Revealing the subkeys of both S_A and S_B reduces the key space by 12 bits.

For S_C and S_D, we discovered that due to the fact of one input bit being constant, there exist *subkey pairs* for which 20 or more out of the 32 possible combinations (5 challenge bits) have the same hamming weight. We observed that in the correlation result it can occur that several distinct subkey candidates lead to correlation values that are very close to each other. We observed that for S_C pairs of two subkey candidates have similar correlation results, where for S_D, there exist pairs of four subkey candidates with this property. Figure 8 shows the correlation over traces for S_D and depicts that phenomenon.

In a worst-case scenario, the CPA on S_C provides 5 bits of information, while attacking S_D only gives 4 bits of information. Correspondingly, in Figure 7, we assume an attack successful if the correct subkey is within the topmost two candidates for S_C and if the correct subkey is within the topmost four candidates for S_D.

According to Figure 7, the overall success probability of recovering 21 bits exceeds 99 % after 300 traces. Our measurement setup requires less than 1.5 seconds to capture one trace, so the attack can be successful with a 99 % probability after 7.5 minutes.

The number of required traces for a successful attack appears surprisingly low to us. A possible explanation for this is the microcontroller itself: The PIC18LC58B only consists of the CPU core. Apart from the GPIO pins, it does not have any peripheral components that would introduce additional noise; the controller is not even equipped with a brown-out detection or interrupt controller.

As a complete CPA result, 21 bits can be revealed in the worst-case scenario. This implies that only 35 bits of the DES keyspace are remaining. That can be used for an exhaustive search of the remaining key space in a very short time: Our test system, a laptop computer with a Intel Core i5-2450M processor running at 2.5 GHz, could search a 32 bit key subspace in 1m58s using a bit-sliced implementation of DES [20]. As a consequence, searching the whole remaining 35 bit key space requires less than 16 minutes on a laptop computer.

5. COUNTERMEASURES

We have shown different ways how the system can be attacked in practice and outlined the attack procedure. We would like to suggest countermeasures that can be deployed for new and existing systems.

5.1 Administrative Options

An attacker's knowledge about a system key gives him access to all locks that are programmed with this key while transponders normally can be programmed for three locking systems, i.e. three different `SID`s with three different keys. Locking system administrators can split large locking systems up into different smaller locking systems. This should be done in accordance with the "need to know" principle, such that each transponder shall only know keys to these areas for which it is intended. In an industry building, this can be, for example,

- one distinct `SID`/key for all external doors
- one distinct `SID`/key for each department
- one distinct `SID`/key for doors to administrative or infrastructural rooms such as server rooms or central heating rooms

With this configuration, an attacker does not get full access to a complete building when a transponder is compromised. For example, if an attacker steals a regular employee

transponder that is authorized for office doors within his department, the attacker cannot enter other departments or unlock external doors. Service staff can, for example, use superordinate keys as described in Section 2.4.2 to get access to more than three locking systems.

This countermeasure is easily implementable without any hardware or software modification. It requires reprogramming transponders and locks once, which is often done in a regular manner anyway. However, the availability of only three locking system slots can make the separation of one big locking system difficult.

The security can be further improved by frequently updating the keys of locking systems in critical locations, such as for external doors or server rooms. This requires a secure update procedure for both locks and transponders. Employees with access to security critical locations can for example, be required to regularly show up in person in order to get a key update. For the updates of locks, it must be ensured that the transmission of key updates is not intercepted.

5.2 Key Derivation

One of the major design problems of this system is the existence of a common system secret that is known to every transponder and every lock in the system. Extraction of this secret by an attacker compromises the entire locking system. As an improvement, one should keep the value of a single device as little as possible and thereby restrict the damage the compromise of a single device can cause.

Starting from a theoretical point of view, using signatures and asymmetric cryptography sounds like a good solution. A central, well protected CA could issue digital certificates to all transponders and locks and also permissions could be embedded in this certificates. These devices could then use those certificates and their private keys for authentication and authorization. However the hardware used in those devices is not capable of executing current asymmetric algorithms with secure key sizes providing a reasonable response time and battery usage.

We would therefore like to introduce the idea of deriving the transponder key from a common system key, which is known to all locks in the system. Using a one way key derivation function, and taking the system key and the transponder ID as input, a transponder key is derived and stored on the transponder. The transponder does not need to know the system key. When a transponder communicates with a lock, the transponder transmits his ID first, and then the lock can also determine the transponder key. If a transponder should ever be lost and all keys are extracted, the attacker can only use those keys to obtain the permissions of this specific transponder, but not of other transponders in the system. Because such a key derivation function can be built using only building blocks from symmetric cryptography, this approach would only need a small amount of additional hardware resources.

SimonsVoss System 3060 Generation 2 (G2) already implements a key derivation scheme that provides each transponder with an individual key, such that only the locks need to know the system-wide key [3].

Also, sufficient key lengths should be chosen for such a system. We currently see 80 bits of symmetric key length as the absolute minimum, but suggest to use 128 bit keys to have security for the foreseeable future. The system we examined was designed almost two decades ago, and a new system should be based on security parameters that can be expected to be secure for at least 20 years and have a reasonable security margin.

5.3 Differential Power Analysis - Hiding and Masking

Hiding and Masking are the two state-of-the art concepts to reduce the leakage that can be exploited by a DPA attack [4, 21]. Hiding aims at reducing the amplitude of the leaking signal while the goal of masking is to make the leaking signal appear as random as possible. Masking can implemented in software at the cost of significantly increasing the memory consumption and execution time.

The PIC16LC58B on the transponder only has 73 bytes of RAM [22]. While we do not know the exact RAM usage while the response is computed, it seems very likely that implementing a masking scheme on that controller would exceed the RAM availability. However, in contrast to the more than 15 year old PIC16LC58B, state-of-the art microcontrollers – or, better – smart card controllers are equipped with enough resources for implementing a masking scheme. Furthermore, the execution time of a masked DES implementation must not exceed the time between transmission of the challenge and expected time of response reception (around 60 ms).

5.4 PRNG Improvement

In Section 4.1 we developed an attack against the random number generator in the locks that is used to generate challenges to transponders. The problem of the current LFSR design is its simplicity, and the low complexity for the attacker to predict PRNG outputs.

Designing and implementing a secure PRNG under the given constraints is a difficult task. We could not find one single solution that offers a high level of security but also retains compatibility and minimizes the effort required to deploy our proposal. Therefore, we split our proposals into three categories.

- **Long-term** modifications require modification of hardware and revising the system concept, while achieving a high level of security.
- **Medium-term** changes are satisfied with software updates of locks and the configuration device.
- **Short-term** improvements shall be implementable by only updating the firmware of the locks.

Considering long-term modifications, it is advisable to use a hardware-based True Random Number Generator, as found on Smart Card and security controllers, instead of – or at least in combination with – a software-based PRNG. An alternative to embedded TRNG circuits can be RF or supply voltage noise. As such components use physical randomness instead of computations, predictions of random numbers are avoided. Furthermore, it is advisable to significantly increase the size of the challenges to render the creation of look-up tables infeasible in practice. We suggest a minimum challenge size of 64 bits.

As a medium-term improvement, we suggest to introduce the ability to update the seed of the PRNG in a secure manner. As an example, the Configuration Devices can be used

to provide additional randomness to the locks during programming cycles. In addition, the short-term proposals shall be included.

As for short-term improvements, all possible sources of randomness shall be considered:

- initial seed, supplied by the manufacturer
- timestamps of successful authentications (configuration devices and transponders)
- *hidden parts of previous challenge/response pairs* (not recommended for security reasons)

Note that it is advisable to avoid deriving PRNG seeds from system secrets. This may give an adversary the opportunity to collect information about the secret.

DES is already available due to the authentication scheme and can be used to actually generate challenges depending on an internal state as well as the entropy sources as mentioned above.

As an example, a random number could be generated using $rand = \text{DES}(\text{key} = mseed \oplus tval, \text{plaintext} = ctr)$, where $mseed$ is a lock-individual random seed programmed by the manufacturer, $tval$ denotes the least significant bits of the timestamps of the previous n successful authentications, and ctr is a counter.

5.5 Protected Hardware

The used microcontroller was susceptible to power glitching and a classic Differential Power Analysis attack. Modern microcontrollers usually have a Brown Out Detection circuitry that makes glitching attacks more difficult. Some controllers also provide hardware support for cryptographic algorithms [23]. However, those protection mechanisms usually are implemented in a low-cost manner and can often be broken without a high effort [24].

In contrast to microcontrollers, the resistance of Smart Card or Secure Access Module (SAM) chips against these and other attacks is a major design goal and such controllers are usually much harder to break than off-the-shelf microcontrollers in terms of equipment and knowledge required as well as attack time. As locking systems are security products par excellence, it seems reasonable for them to use modern security controllers.

Surprisingly, we do not know any widely spread battery-powered locking systems that use smart card or SAM controllers, or deploy any other countermeasures against hardware attacks.

6. PRACTICAL RELEVANCE

Besides the theoretical results we showed in the paper, we would like to discuss the practical relevance and importance of our findings in this section.

The security of mechanical locks has been examined for more than a century and requirements for mechanical locks have been standardized. A very common standard for the security of mechanical locks is VdS 2156-1 [25] which defines multiple levels of security for mechanical locks. A lock has to resist against an attacker who uses lockpicking tools for at least 3 or 6 minutes, depending on the required security level. In addition to that, high security locks must use keys, that cannot be cloned easily.

All attacks we presented need a total attack time of more than 6 minutes. Also the transponders have a protection against unauthorized cloning which however can be defeated using the brute force attack or the side channel attack in Section 4. According to our understanding, this would not prevent a high security certification according to VdS 2156-1.

However, there are important differences between the attacks we presented, and typical attacks on mechanical locks:

- To attack a mechanical lock, one usually needs to be with the lock during the attack, and this can be easily noticed by other people near the lock. Also it could leave non removable traces on the lock itself. Our proposed attacks might remain unnoticed.
- Our total attack time when cloning a transponder or escalating privileges is longer than 6 minutes, while the time spent with a lock is even less than 3 minutes. However, this is more a key cloning attack than an attack on the lock itself.
- While picking locks or cloning mechanical keys usually requires some human skills to operate the equipment properly, the attacks we presented can be heavily automatized and afterwards be done by persons with no or little training in picking locks.

For electronic systems, there exists the additional standard VdS 2156-2 [26] with more specific requirements. The standard describes *intelligent attacks*, and a lock needs to withstand such an attack for 10 to 90 minutes, depending on the security level. The standard also describes requirements for wireless access tokens:

- The transmission of the "opening code" must be *encrypted*. However, no keylength or other details of the used encryption scheme are specified.
- The "code" may only be transmitted upon an explicit action, and third parties may not read out the transponder or determine usable information.

It is our understanding that [26] only aims at outsider attacks where an adversary is not in possession of a transponder knowing the system secret of a lock being attacked. Therefore, we assume that our publication does not affect the compliance of the SimonsVoss System 3060 G1 locking system to this standard. While a compliance with [25, 26] may be helpful from an insurance and liability perspective, we strongly believe that especially [26] is not sufficient for ensuring state-of-the-art security of electronic locks. In particular, we see the following weaknesses in this standard:

- While the transmission shall be "encrypted", no requirements are given with respect to the cryptographic primitives to be used. We suggest requiring accepted and strong cryptographic algorithms with a sufficiently large key space, as well as a proper design of the cryptographic protocols.
- While the protection against "intelligent attacks" is required, that term is not properly defined. In our notion, this could mean anything from brute-force attacks over the exploitation of weaknesses in the cryptographic algorithms and protocols up to side-channel or fault attacks. The range of intelligent attacks is very broad and reducing this whole spectrum to one term

makes the comparison between different products difficult. We suggest properly defining requirements for distinct attack types.

- The notion of attack time in the area of electronic locks is ambiguous. For mechanical attacks, it is usually clear that attack time refers to the time the attacker must be physically close to the lock. We propose to clearly distinguish between *access token availability time* in which the transponder or other access token must be in the possession of the attacker, the *lock availability time* in which the attacker must be physically close to the clock and the *computation time* in which neither access token nor lock needs to be accessible by the attacker.

In addition to those suggestions, we strongly recommend to define testing procedures in order to allow independent compliance verification.

In its current version, the VdS standards do not motivate the locking system industry to enforce state-of-the-art electronic security in their products.

The German Federal Office for Information Security (BSI, Bundesamt für Sicherheit in der Informationstechnik) fills this gap by issuing more detailed technical guidelines about the security of electronic identification and access control systems. These guidelines are primarily important for governmental organizations, but also may be helpful for security aware institutions that need to select appropriate products.

TR-03126-5 [27] gives a detailed analysis of RFID systems for the generic use case of "employee identification". It describes the system architecture and components of RFID systems, gives examples of more concrete use cases, discusses security requirements and points out possible attack vectors and countermeasures. TL-03405 [28] defines security requirements and test criteria concretely for electronic locking systems, and TL-03424 [29] defines the requirements for electronic keys in particular.

The requirements defined by [27, 28, 29] are comprehensive and go down to a detailed technical level, for example, by defining the cryptographic algorithms to be used. Examples of such requirements are

- 3-DES, AES or cryptographic algorithms with a comparable security level shall be used to protect transmissions.
- Secret keys shall be stored in a secure manner, for example by using Secure Authentication Modules (SAM).
- A system and its devices shall be certified at least to Common Criteria EAL 3 [30].
- The secret keys of replaced or lost components shall be declared invalid unless it is ensured that the keys cannot be read out.

To conclude our standards review, relying on the VdS certifications may lead to a false sense of security on the customer side in spite of the fact that many locking systems, besides SimonsVoss, are potentially vulnerable. We believe that the requirements from the BSI standards are a good approach towards designing secure locking systems, and that products designed or even certified to the current BSI guidelines may provide higher security. However, there do not exist many BSI-certified products, we even have not found any major locking system manufacturer advertising protection against particular hardware attacks, such as side-channel attacks.

From an attacker's point of view, our Brute Force and Power Analysis attacks have the advantage to mechanical attacks that they can hardly be discovered: The only trace the attacker potentially leaves is an entry in the access log of the lock, however logging is a feature which is only available in more expensive model variants of locks. If logging is enabled, the attacker can download the access log and select a `TID` to impersonate that causes the least suspicion. This cannot only disguise the intrusion, but also lead to a false accusation of non-involved persons in the case that suspicious activities are discovered. While this type of attack seems too elaborate for small-time criminals, we believe that it is a realistic threat with respect to large-scale attacks such as espionage, intelligence operations or organized crime.

7. CONCLUSION

In this paper, we have reverse engineered and analyzed the SimonsVoss System 3060 Generation 1 electronic locking system, and we determined the level of difficulty for a successful attack. SimonsVoss products are used to protect sensitive infrastructures. We found an attack against the PRNG in our analyzed G1 locks, and we showed that the G1 system concept uses one master secret that is known to all devices of a particular locking system. We explained that this secret can be extracted by eavesdropping on transponder authentication runs followed by a brute-force attack, or by conducting a Differential Power Analysis attack against a transponder. We proposed countermeasures against all discussed attacks.

We also examined security standards for electronic locks and found out that the VdS standards might not be sufficient to enforce a high level of security. It is our impression that this gap may lead the locking system industry in a wrong direction; we would not be surprised about other products from other manufacturers that show electronic vulnerabilities as well.

What we did not evaluate in this paper is the surrounding environment that also protects critical installations besides the locks. Attacking a lock is just one out of many ways to gain access to a building or room. Other methods may be attacks on the door or windows of the room itself, without attacking the lock directly. Stealing a valid transponder or bribing/blackmailing/impersonating somebody who has access to the room could also be a successful attack strategy. We still think that most attackers aiming at breaking into a SimonsVoss System 3060 G1-secured facility will choose another way than using a cryptographic attack against the door lock – a system is only as secure as the weakest link in the chain.

Acknowledgements

This work was partly funded by the German Federal Ministry of Education and Research (BMBF) in the project SIBASE through grant number 01S13020A and within the project EC SPRIDE and also by the Hessian LOEWE excellence initiative within CASED. We would also like to thank Jan Morawek, starbug and krikkit from the Chaos Computer Club in Germany for their ideas and support.

8. REFERENCES

[1] M. W. Tobias, M. Fiddler, and T. Bluzmanis, "Invisible Access - Opening New Doors to Insecurity," DEFCON 17, 2009, http://defcon.org/images/defcon-17/dc-17-presentations/defcon-17-marc_weber_tobias-matt_fiddler-invisible_access.pdf, accessed on 26.07.2013.

[2] M. Kasper, T. Kasper, A. Moradi, and C. Paar, "Breaking keeloq in a flash: On extracting keys at lightning speed," in *Progress in Cryptology – AFRICACRYPT 2009*, ser. Lecture Notes in Computer Science, B. Preneel, Ed. Springer Berlin Heidelberg, 2009, vol. 5580, pp. 403–420. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02384-2_25

[3] D. Strobel, B. Driessen, T. Kasper, G. Leander, D. Oswald, F. Schellenberg, and C. Paar, "Fuming acid and cryptanalysis: Handy tools for overcoming a digital locking and access control system," in *Advances in Cryptology – CRYPTO 2013*, ser. Lecture Notes in Computer Science, R. Canetti and J. Garay, Eds. Springer Berlin Heidelberg, 2013, vol. 8042, pp. 147–164. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40041-4_9

[4] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology — CRYPTO' 99*, ser. Lecture Notes in Computer Science, M. Wiener, Ed. Springer Berlin Heidelberg, 1999, vol. 1666, pp. 388–397. [Online]. Available: http://dx.doi.org/10.1007/3-540-48405-1_25

[5] S. Spitz, M. Pramateftakis, and J. Swoboda, *Kryptographie und IT-Sicherheit*. Wiesbaden, Germany: Vieweg + Teubner Verlag / Springer Fachmedien, 2011.

[6] J. Weyers, "Showing your keys on TV: What could possibly go wrong?" 2013, https://program.ohm2013.org/event/49.html, accessed on 16.08.2013.

[7] A. Kerckhoffs, "La cryptographie militaire," *Journal des sciences militaires*, 1883.

[8] F. Garcia, G. Koning Gans, R. Muijrers, P. Rossum, R. Verdult, R. Schreur, and B. Jacobs, "Dismantling mifare classic," in *Computer Security - ESORICS 2008*, ser. Lecture Notes in Computer Science, S. Jajodia and J. Lopez, Eds. Springer Berlin Heidelberg, 2008, vol. 5283, pp. 97–114. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-88313-5_7

[9] H. Plötz and K. Nohl, "Peeling away layers of an rfid security system," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, G. Danezis, Ed. Springer Berlin Heidelberg, 2012, vol. 7035, pp. 205–219. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-27576-0_17

[10] "SimonsVoss Technologies AG: Historie," http://www.simons-voss.de/Historie.32.0.html, accessed on 24.07.2013.

[11] "SECURITY 2006 Messe News," 2006, http://web.archive.org/web/20061211054124/http:/www.simons-voss.de/fileadmin/media/home/Internet_Flyer_deutsch.pdf, accessed on 26.08.2013.

[12] "Handbuch LSM - Benutzer," Tech. Rep., Jul 2010, http://www.simons-voss.de/fileadmin/php/fileadmin/downloads/ger/lsm/HB_LSM_30_Benutzer_V1.0_D.pdf, accessed on 26.07.2013.

[13] N. F. PUB, "46-3. data encryption standard," *Federal Information Processing Standards, National Bureau of Standards, US Department of Commerce*, 1977.

[14] bunnie, "Hacking the PIC 18F1320," 2007, http://www.bunniestudios.com/blog/?page_id=40, accessed on 26.07.2013.

[15] "Unmarked Die Revisions :: Part I," 2007, http://www.flylogic.net/blog/?p=9, accessed on 26.07.2013.

[16] "CloudCracker - Dictionaries," https://www.cloudcracker.com/dictionaries.html, accessed on 13.08.2013.

[17] K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic analysis: Concrete results," in *Cryptographic Hardware and Embedded Systems — CHES 2001*, ser. Lecture Notes in Computer Science, Koç, ÇetinK. and Naccache, David and Paar, Christof, Ed. Springer Berlin Heidelberg, 2001, vol. 2162, pp. 251–261. [Online]. Available: http://dx.doi.org/10.1007/3-540-44709-1_21

[18] J. H. Stewart, "Future testing of large LSI circuit cards," in *Semiconductor Test Symposium*. IEEE, 1977, pp. 6–15.

[19] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems - CHES 2004*, ser. Lecture Notes in Computer Science, M. Joye and J.-J. Quisquater, Eds. Springer Berlin Heidelberg, 2004, vol. 3156, pp. 16–29. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-28632-5_2

[20] Matthew Kwan, "Bitslice DES," http://www.darkside.com.au/bitslice/, accessed on 12.08.2013.

[21] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[22] "PIC16C5X Data Sheet," Microchip Technology Inc., Tech. Rep., 2002, http://ww1.microchip.com/downloads/en/devicedoc/30453d.pdf, accsessed on 26.07.2013.

[23] "PIC16C5X Data Sheet," Microchip Technology Inc., Tech. Rep., 2002, http://ww1.microchip.com/downloads/en/devicedoc/30453d.pdf, accessed on 12.08.2013.

[24] I. Kizhvatov, "Side channel analysis of avr xmega crypto engine," in *Proceedings of the 4th Workshop on Embedded Systems Security*, ser. WESS '09. New York, NY, USA: ACM, 2009, pp. 8:1–8:7. [Online]. Available: http://doi.acm.org/10.1145/1631716.1631724

[25] "VdS-Richtlinien für mechanische Sicherungseinrichtungen - Schließzylinder mit Einzelsperrschließung," VdS, Tech. Rep., 2012.

[26] "VdS-Richtlinien für mechanische Sicherungseinrichtungen - Schließzylinder mit Einzelsperrschließung - Teil 2: Elektronische Schließzylinder," VdS, Tech. Rep., 2012.

[27] "BSI TR-03126-5, Technische Richtlinie für den sicheren RFID-Einsatz (TR RFID), Einsatzgebiet

elektronischer Mitarbeiterausweis," Bundesamt für Sicherheit in der Informationstechnik, Tech. Rep., 2010.

[28] "BSI TL-03405, Anforderungen und Prüfbedingungen für elektronische Schließzylinder und Schließsysteme," Bundesamt für Sicherheit in der Informationstechnik, Tech. Rep., 2010.

[29] "BSI TL-03424, Ergänzung zu BSI TL elektronische Schließsysteme, Zutrittskontrollanlagen; Anforderungen für elektronische Schlüssel," Bundesamt für Sicherheit in der Informationstechnik, Tech. Rep., 2011.

[30] "Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components," Tech. Rep., 2012, https://www.niap-ccevs.org/Documents_and_Guidance/cc_docs.cfm, accessed on 25.08.2013.

APPENDIX

Additional Figures

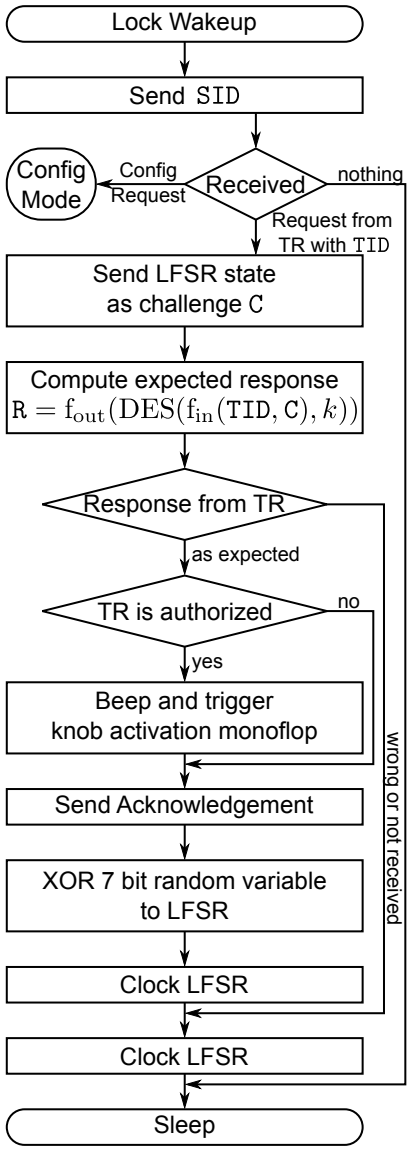

Figure 9: LK behavior during TR authentication