Physical Cryptanalysis of KeeLoq Code Hopping Applications

Thomas Eisenbarth[1], Timo Kasper[1], Amir Moradi[2,⋆], Christof Paar[1],
Mahmoud Salmasizadeh[2], and Mohammad T. Manzuri Shalmani[2]

[1] Horst Görtz Institute for IT Security
Ruhr University of Bochum, Germany
[2] Department of Computer Engineering and Electronic Research Center
Sharif University of Technology, Tehran, Iran
{eisenbarth,kasper,moradi,cpaar}@crypto.rub.de
{salmasi,manzuri}@sharif.edu

Abstract. Recently, some mathematical weaknesses of the KeeLoq algorithm have been reported. All of the proposed attacks need at least 2^{16} known or chosen plaintexts. In real-world applications of KeeLoq, especially in remote keyless entry systems using a so-called code hopping mechanism, obtaining this amount of plaintext-ciphertext pairs is rather impractical. We present the first successful DPA attacks on numerous commercially available products employing KeeLoq code hopping. Using our proposed techniques we are able to reveal not only the secret key of remote transmitters in less that one hour, but also the manufacturer key of receivers in less than one day. Knowing the manufacturer key allows for creating an arbitrary number of valid transmitter keys.

1 Motivation

In 1999, Kocher *et. al* [5] proposed several methods for analyzing the information leakage of implementations of security related systems. The most powerful attack in this area is called DPA (Differential Power Analysis) and exploits power consumption traces of cryptographic hardware to reveal confidential information. Almost ten years later, DPA remains an attack mostly performed in smart card evaluation labs and universities, only targeting their own known implementations.

In this paper we present two DPA attacks on KeeLoq real-world applications. They target hardware and software implementations of which we had no prior knowledge about details and design architecture. The attacks can reveal the device key of a particular transmitter and also the manufacturer key used in a receiver in one hour or one day, respectively. Thus, it is possible to clone an existing transmitter or to generate an arbitrary number of valid device keys with the manufacturer key, such that the transmitters produced with these appear to be authentic.

[⋆] Amir Moradi performed most of the work described in this contribution as a visiting researcher at the Ruhr-University of Bochum.

In Section 2, previous mathematical and theoretical attacks on the KeeLoq algorithm and their difficulties in real-world applications are briefly described. Furthermore, we denote the key derivation schemes we are aware of. Section 3 gives details about our attacks and Section 4 concludes the paper.

2 Previous Work

The first two attacks on KeeLoq were published by Bogdanov in [2]. One attack is based on slide and guess-and-determine techniques and needs about $2^{50.6}$ KeeLoq encryptions. The other one additionally uses a cycle structure analysis technique and requires 2^{37} encryptions. However, both attacks require all 2^{32} plaintext-ciphertext pairs.

Afterwards, Courtois *et. al* [4] proposed two attacks. One is a slide-algebraic attack demanding for 2^{53} KeeLoq encryptions and 2^{16} known plaintexts. The second attack uses a cycle technique similar to the above and can be carried out knowing nearly 2^{32} known plaintexts. It reveals the secret key with a complexity of approximately 2^{29} KeeLoq encryptions.

Recently, Biham *et. al* presented a brief description of another successful attack on the KeeLoq algorithm [1]. It requires 2^{16} chosen plaintexts and can find the secret key in two days using 50 Dual Core machines.

As mentioned in [2], the above attacks are appropriate for KeeLoq IFF (Identify Friend or Foe) systems because it is theoretically possible to collect all 2^{32} plaintext-ciphertext pairs within about 100 days from a commercial KeeLoq IFF system. Furthermore, it would be easily possible to collect 2^{16} chosen plaintexts in about one hour [1]. However, none of these attacks works on applications employing the KeeLoq code hopping technique, because of a discrimination value (a 12-bit or 10-bit fixed part of plaintexts[1]) that is not known to the attacker. To our knowledge, most of the commercial implementations of KeeLoq as a remote keyless entry system employ the code hopping mechanism. Thus, the described attacks are not considered as a big threat for their security.

Moreover, in [1] it is claimed that finding one KeeLoq key leaks the manufacturer secret. This is correct if and only if the secret key of the transmitter is obtained by applying an XOR to the manufacturer key and a specification of the device (for instance a serial number or a seed value of the transmitter). In contrast, we know several companies that use other key derivation schemes. For instance, the device key is obtained by decrypting a specification of the device using the manufacturer key. In this case, finding the secret device key of a KeeLoq transmitter does not lead to reveal the manufacturer key. For clarification, the known key derivation schemes are reviewed in the following:

1. The device key k_{dev} is obtained by two KeeLoq decryptions. Two functions of the device serial number (which are usually simple padding) generate the plaintexts for the decryptions.

$$k_{dev} = dec_{k_M}(PAD1||S/N)||dec_{k_M}(PAD2||S/N)$$

[1] See [2] for more information about the structure of hopping codes.

Note that the serial number is known, because it follows the ciphertext in each transmitted message.

2. Another key derivation scheme is similar to the previous one except for a randomly generated seed value which is stored in the transmitter and is used to generate the device key. Having physical access to the transmitter it can be forced to send its seed value.
3. Sometimes, the device key k_{dev} is generated from an XOR of a simple function of the device serial number with the manufacturer key k_M.

$$k_{dev} = k_M \oplus (PAD1||S/N||PAD2||S/N)$$

If the attacker reveals a device key, the manufacturer key is computed easily.

4. The last scheme is similar to the third one. The device key is derived from an XOR of the manufacturer key and a simple function of the serial number of the device and its seed value. The attacker can find the manufacturer key if physical access to the transmitter is given.

Note that a manufacturer may develop a proprietary key derivation scheme not included in the above list.

3 Our Attack Scenarios

We introduce two DPA attacks on KeeLoq code hopping systems. The first reveals the secret device key from an integrated circuit that performs the encryption in a transmitter. The second attack is executed on the receiver to recover the manufacturer key from a software implementation running on a microcontroller. The details of the attacks follow.

3.1 DPA Attack on Transmitter

In order to perform a DPA attack, the attacker should have some knowledge about the architecture and details about the targeted device. We did not have any information about the architecture and design detail of commercial products of KeeLoq code hopping encoders. The first problem we encountered was finding the points in time of the power consumption traces that correspond to the encryption function. We were able to find it using statistical methods after several thousand measurements. In addition, the commercial KeeLoq code hopping encoders use an internal RC oscillator as clock generator. Thus, there is a strong jitter in the power consumption traces leading to misalignment which severely affects the triggering and attack processes. Also, the post-processing of several thousand power traces, each containing 8 million samples, was a time-consuming task.

By analyzing the power traces, we found out that there is a specific hardware inside the chip to perform the KeeLoq encryption. Next we analyzed the KeeLoq encryption algorithm to model its power consumption. KeeLoq consists of two shift registers[2]. One of the registers rotates the key bits. The other

[2] See [2] for more information about the KeeLoq algorithm.

one is an NLFSR (non-linear feedback shift register) storing the 32-bit state. It is well-known that a CMOS flip-flop consumes significantly more power if its state toggles than if its value remains constant. Hence, a Hamming distance model is appropriate for describing the power consumption. Note that the Hamming distance of the shift register rotating the key bits does not change during the 528 encryption rounds. This leads to a constant power consumption of the key register in each clock cycle. Therefore it is not possible to find a correlation between key bits in the register and power traces.

Instead, we focused on the state shift register (NLFSR). Since we do not have access to the plaintext but to the ciphertext only, we analyzed the information leakage in the reverse direction, *i.e.*, starting in the 528th round with the ciphertext and working towards the plaintext, by executing a correlation DPA attack [3]. We first hypothesized one bit of the key at a time. It turned out that recovering only one bit in each round of the attack was impossible, because the biggest difference in the Hamming distance of two consecutive contents of the state shift register is one. Thus, we enhanced the attack strategy by hypothesizing eight bits of the key and recovering four of these bits in each round of the attack.

We performed this attack on several chips with different part numbers in DIP or SOIC packages. We are able to recover the secret key of KeeLoq encoders in DIP packages from only 10 power traces. Clearly, SOIC packages benefit from a smaller process technology so the power consumption values are smaller than DIP packages. Hence, the SNR (signal-to-noise ratio) is decreased and we need more power traces. Still, at most 50 power traces are sufficient to reveal the secret key of a device in an SOIC package. Collecting the power traces and finding the secret key is performed in less that one hour. Note that, with respect to the fastest existing attack [1], this is even less than the time required to collect 2^{16} chosen plaintexts. Also, noise of the measurement setup and the sampling rate significantly affect the efficiency of our attack.

3.2 DPA Attack on Receiver

Generally, code hopping decoders (receivers) are implemented in software on microcontrollers because they can be more flexibly adapted to different learning and key derivation schemes than ASIC chips. Similar to our first attack on the transmitter, we did not know implementation details of the receiver. We just know the type of microcontroller (as it is often printed on the chip package) and its instruction set. We assume that shift instructions are used to implement the state and key shift registers of the KeeLoq decryption algorithm. Moreover, we did not know whether the receiver stores the key of each authenticated transmitter or uses a key derivation process to generate each device key during normal operation (*i.e.* during the authentication process). Since the manufacturer key is only used during the key derivation process, we had to find the point in time when the key derivation process is executed. Since every receiver has to use the key derivation routine during learning phase, we acquired power traces of this phase. For this, we developed a simple device that emulates a code hopping encoder and is capable of sending authentic hopping codes with different serial numbers and seed values via the RF(Radio Frequency) interface. With this emulator we efficiently collected several thousand power traces of the learning phase with random serial numbers generated by us.

In order to perform the attack, we characterized the power leakage of shift instructions of the employed 8-bit microcontroller. As with most microcontrollers, it leaks the Hamming weight of operands. Again, we only knew the ciphertexts that were generated from the random serial numbers, but not the plaintexts. Similar to our first described attack, where the DPA on the encryption was carried out in the order from the last to the first round, we want to find the secret key of the decryption starting from the first round. Thus, parts of the first attack could be reused with a modified power model. Analyzing the acquired power traces, we found that the correlation coefficient between the hypothetical power values and the power traces decreases with the number of rounds of the attack. The reason for this increasing misalignment turned out to be a dependency of the number of clock cycles needed for one round of the algorithm on the data being processed. Finally, due to some optimizations with regard to these timing issues, we could reveal the manufacturer key using 1000 power traces. The required time to collect the power traces and to perform the attack is less than one day. Note that the employed key derivation scheme used in the attacked receiver is equivalent to the first key derivation method, discussed in Section 2. Accordingly, the serial number is padded by a fixed value before being decrypted using the manufacturer key to obtain a part of the device key. We verified the revealed manufacturer key by programming a virgin code hopping encoder with a random serial number and authenticating it to the receiver.

4 Conclusion

Although some theoretical attacks on the KeeLoq algorithm have recently been reported, none of them is able to break the code hopping systems in a reasonable time. We illustrated the difficulties of those attacks in the presence of different key derivation schemes.

In this paper we presented the first successful practical attacks on KeeLoq code hopping systems. A DPA attack performed on the hardware chip of a transmitter reveals the device key in less than one hour. Another DPA targeting a software implementation running on an 8-bit microcontroller inside a receiver allows to recover the full manufacturer key in less than one day. Note that we did not have any prior knowledge about architecture and design details of the attacked commercial KeeLoq code hopping encoders and decoders. These very effective attacks represent a real practical threat for many commercial applications employing the KeeLoq algorithm.

References

1. E. Biham, O. Dunkelman, S. Indesteege, N. Keller, and B. Preneel. How to Steal Cars – A Practical Attack on KeeLoq. CRYPTO 2007 Rump Session. `http://www.`

cosic.esat.kuleuven.be/keeloq/.

2. A. Bogdanov. Attacks on the KeeLoq Block Cipher and Authentication Systems. In *3rd Conference on RFID Security 2007 (RFIDSec 2007)*, 2007. http://www.crypto.rub.de/imperia/md/content/texte/publications/conferences/keeloq_rfidsec2007.pdf.
3. E. Brier, C. Clavier, and F. Olivier. Correlation Power Analysis with a Leakage Model. In M. Joye and J.-J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 16–29. Springer, 2004.
4. N. T. Courtois, G. V. Bard, and D. Wagner. Algebraic and slide attacks on keeloq. Cryptology ePrint Archive, Report 2007/062, 2007. http://eprint.iacr.org/.
5. P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *CRYPTO '99: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, pages 388–397, London, UK, 1999. Springer-Verlag.