Algebraic Methods in Side-Channel Collision Attacks and Practical Collision Detection

Andrey Bogdanov[1], Ilya Kizhvatov[2], and Andrey Pyshkin[3]

[1] Horst Görtz Institute for Information Security
Ruhr-University Bochum, Germany
abogdanov@crypto.rub.de
[2] University of Luxembourg, Luxembourg
ilya.kizhvatov@uni.lu
[3] Technical University Darmstadt, Germany
pychkine@cdc.informatik.tu-darmstadt.de

Abstract. This paper presents algebraic collision attacks, a new powerful cryptanalytic method based on side-channel leakage which allows for low measurement counts needed for a successful key recovery in case of AES. As opposed to many other side-channel attacks, these techniques are essentially based on the internal structure of the attacked cryptographic algorithm, namely, on the algebraic properties of AES. Moreover, we derived the probability distributions of Euclidean distance for collisions and non-collisions. On this basis, a statistical framework for finding the instances of side-channel traces leaking most key information in collision attacks is proposed.
Additionally to these theoretical findings, the paper also contains a practical evaluation of these side-channel collision attacks for a real-world microcontroller platform similar to many smart card ICs. To our best knowledge, this is the first real-world study of collision attacks based on generalized internal collisions. We also combined our methods with ternary voting [1] which is a recent multiple-differential collision detection technique using profiling, where neither plaintexts, ciphertexts nor keys have to be known in the profiling stage.

Key words: Side-channel attacks, collision attacks, algebraic cryptanalysis, multiple-differential collision attacks, ternary voting, AES, DPA

1 Introduction

Motivation. The motivation of this paper is to develop a framework minimizing the number of online measurements needed for a successful key recovery in a real-world noisy environment. This is to a certain extent equivalent to extracting a maximum amount of key information from the given side-channel signal, which is the central question of side-channel cryptanalysis. In practice, this setting is important in such cases where the attacker has very restricted access to the device due to organizational policies or where only few cryptographic operations

with the same key are allowed, which is used as a side-channel countermeasure in some real-world systems. An independent line of motivation we pursue is to come up with an efficient and practical alternative to such well-known side-channel techniques as differential power analysis (DPA) [2] and template attacks [3], [4] which would be free of their main natural limitations: Dependency on a certain leakage model for DPA and the necessity of thoroughly characterizing the attacked device for template attacks.

Side-channel *collision attacks* are a well-suited base for the solution of these problems due to the inherently low numbers of needed measurements, the absence of any concrete leakage model, and the possibility to build collision templates without detailed knowledge of the target.

Collision attacks. Basic side-channel collision attacks [5] were improved in [6] by introducing the notion of *generalized collisions* that occur if two S-boxes at some arbitrary positions of some arbitrary rounds process an equal byte value within several runs. However, [6] treats only the *linear collisions* of AES which are generalized collisions that occur in the first AES round only. Moreover, the results in [6] as well as those in [5] assume that the collision detection is absolutely reliable, while there are significant error probabilities in real-world scenarios. Though this problem was approached in [1] by introducing multiple-differential collision detection (binary and ternary voting), a sound real-world evaluation of these methods is still lacking.

Our contribution. Additionally to linear collisions, we consider *nonlinear collisions* that are defined as generalized collisions comprising several rounds. They deliver extra information contained in further AES rounds. Each such collision can be considered as a nonlinear equation over a finite field. The set of all detected collisions corresponds to a system of nonlinear equations with respect to the key, which can be solved using techniques closely related to the algebraic cryptanalysis of AES with a reduced number of rounds which are referred to as *algebraic collision-based key recovery*.

For collision detection, the Euclidean distance is used. We obtain probability distributions of this statistic in the univariate Gaussian noise model. We show that for large numbers of points in the side-channel trace these two Euclidean distances can be approximated by normal distributions with different parameters. This allows us to define a statistical metric for the time instants of the trace leaking most information for collision detection. It turns out that these points are quite different from those leaking key data in standard DPA.

Combining these improvements, we achieve a considerable reduction of the number of online measurements needed for a successful key recovery. We implemented the attacks for an Atmel AVR ATMega16 microcontroller. The practical results can be found in Table 1. In a version of the attack, we additionally use collisions from ternary voting, a multiple-differential collision detection technique from [1]. Neither plaintexts, ciphertexts nor keys have to be known in the profiling stage.

This indicates that the algebraic collision attacks on AES without profiling are superior to standard CPA in terms of number of measurements needed. Rather surprisingly, the efficiency of our collision techniques *without profiling* is comparable to the stochastic methods [4] *with profiling* (one of best known template-based attacks) for low numbers of profiling curves. Moreover, if profiling is allowed for collision attacks, the number of online measurements can be further reduced. Note that all the implemented collision techniques (both with and without profiling) do use the knowledge of the time instances leaking most information.

Table 1. Summary of results: Hamming-distance based CPA, basic collision attack (on our ATmega16 AES implementation) without profiling and stochastic methods with profiling (on an ATM163 AES implementation [4]) vs. collision attacks based on FL-collisions with and without profiling for $C_{\text{offline}} \leq 2^{40}$ (P – success probability, C_{online} – number of online measurements, $C_{\text{profiling}}$ – number of profiling measurements, C_{offline} – number of offline operations for key recovery)

	P	C_{online}	$C_{\text{profiling}}$
HD-based CPA	0.8	61	0
Basic collision attack [5]	0.85	300	0
Stochastic methods [4] for ATM163	0.82	10	200
FL-collisions, this paper	0.76	16	0
FL-collisions, this paper	0.72	12	625

2 Preliminaries

2.1 Basic Notation

All collision attacks have two stages: an *online stage*, where measurements on the target device implementing the attacked cryptographic algorithm are performed, and an *offline stage*, where the cryptographic key is obtained from the traces acquired in the online stage. Additionally, a collision attack can be enhanced to have a *profiling stage*, where some profiling traces are obtained from some implementation of the attacked cryptographic algorithm.

In this paper we perform our collision attacks at the example of AES. We use the following notation to represent its variables. $K = \{k_j\}_{j=1}^{16}$, $k_j \in \text{GF}(2^8)$ is the 16-byte user-supplied key (the initial AES subkey). $X = \{x_j\}_{j=1}^{16}$, $Y = \{y_j\}_{j=1}^{16}$ and $Z = \{z_j\}_{j=1}^{16}$, $x_j, y_j, z_j \in \text{GF}(2^8)$ are the first, next to the last and last 16-byte AES subkeys, respectively. AES plaintexts are denoted by $P^i = \{p_j^i\}_{j=1}^{16}$, $p_j^i \in \text{GF}(2^8)$ and ciphertexts by $C^i = \{c_j^i\}_{j=1}^{16}$, $c_j^i \in GF(2^8)$, where $i = 1, 2, \ldots$ is the number of AES execution.

Collision-based key recovery methods for AES are mainly parametrized by the number γ of random plaintexts and/or ciphertexts needed to obtain the cryptographic key with success probability P. In our collision attacks, γ can be chosen between 4 and 20 in the majority of cases. We are interested in success probabilities $P \geq 0.5$.

2.2 Linear Collision-Based Key Recovery

Given a linear collision (within the first round of AES), one obtains a linear equation with respect to the key over GF(2^8) of the form

$$S(p_{j_1}^{i_1} \oplus k_{j_1}) = S(p_{j_2}^{i_2} \oplus k_{j_2}), \text{ or } k_{j_1} \oplus k_{j_2} = p_{j_1}^{i_1} \oplus p_{j_2}^{i_2} = \Delta_{j_1,j_2} \text{ for } j_1 \neq j_2.$$

In the example of Figure 1, one has the following equation: $k_4 \oplus k_{11} = p_4^1 \oplus p_{11}^2 = \Delta_{4,11}$. S-boxes where collisions occurred (*active* S-boxes) are marked by the numbers of collisions they account for. The input byte p_j^i is characterized by its position $j \in \{1, \ldots, 16\}$ within the plaintext block and the number $i = 1, 2, \ldots$ of the plaintext block it belongs to.

If D collisions have been detected, they can be interpreted as a system of linear binomial equations over GF(2^8):

$$\begin{cases} k_{j_1} \oplus k_{j_2} = \Delta_{j_1,j_2} \\ \ldots \\ k_{j_{2D-1}} \oplus k_{j_{2D}} = \Delta_{j_{2D-1},j_{2D}} \end{cases}$$

This system cannot have the full rank due to the binomial form of its equations. Moreover, for small numbers of inputs to AES the system is not connected and it can be divided into a set of h_0 smaller independent (with disjunct variables) connected subsystems with respect to the parts of the key. Each subsystem has one free variable. Let h_1 be the number of all missing variables, and $h = h_0 + h_1$. Then the system has 2^{8h} solutions. That is, $C_{\text{offline}} = 2^{8h}$ guesses have to be performed, which is the offline complexity of the attack. Each key hypothesis is then tested using a known plaintext-ciphertext pair to rule out incorrect candidates. C_{offline} quickly becomes feasible as the number of distinct inputs grows. The probability that $C_{\text{offline}} \leq 2^{40}$ ($h \leq 5$) is about 0.85 for $\gamma = 6$, if all collisions are detected. Note that the question of reliable collision detection was not treated in [6].

2.3 Direct Binary Comparison Using Side-Channel Signal

There are ways of deciding if two S-boxes accept equal inputs using side-channel information obtained from the *implementation* of the attacked cryptographic algorithm. For instance, typical side channels are the power consumption of devices as well as their electromagnetic radiation, which manifest some data and key dependency in many cases.

Given two side-channel traces $\tau_1 = (\tau_{1,1}, \ldots, \tau_{1,l}) \in \mathbb{R}^l$, $\tau_2 = (\tau_{2,1}, \ldots, \tau_{2,l}) \in \mathbb{R}^l$, respectively corresponding to some pair of S-box executions with inputs a_1

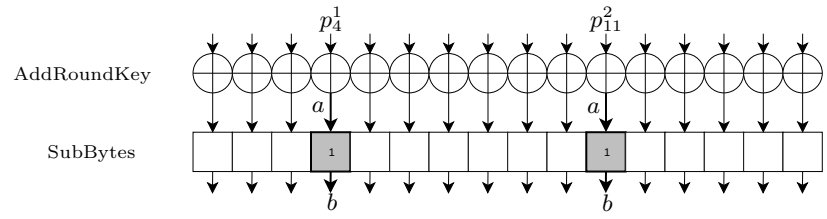

Fig. 1. A linear collision for some pair of runs

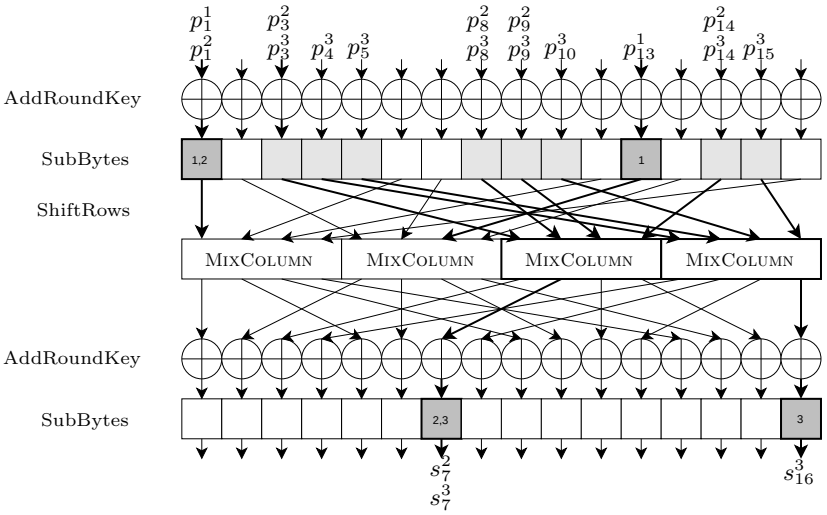

Fig. 2. FS-collisions

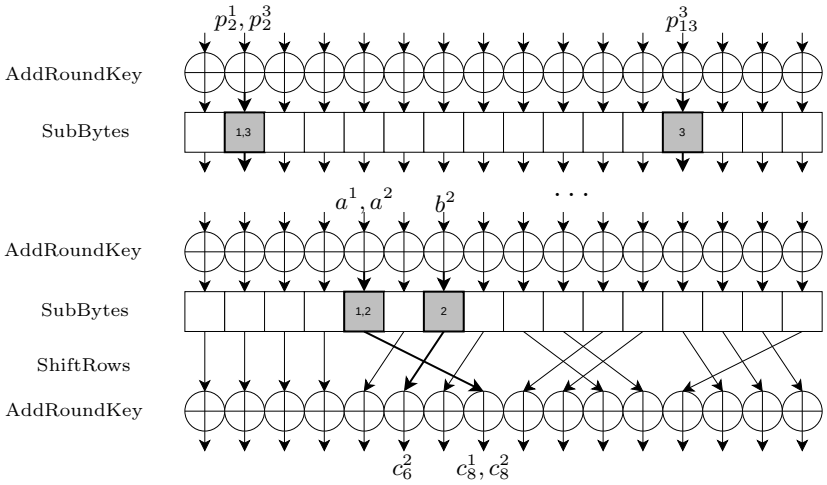

Fig. 3. FL-collisions

and a_2, it has to be decided whether $a_1 = a_2$ for collision detection. In this paper we use the Euclidean distance based binary comparison test T (as in [5] and [1]) for this purpose:

$$T(\tau_1, \tau_2) = \begin{cases} 0 \text{ (no collision)}, & \text{if } H(\tau_1, \tau_2) < W \\ 1 \text{ (collision)}, & \text{if } H(\tau_1, \tau_2) \geq W, \end{cases}$$

where W is a decision threshold and H is the following statistic:

$$H(\tau_1, \tau_2) = 1/\sum_{j=1}^{l} (\tau_{1,j} - \tau_{2,j})^2.$$

Test T is characterized by the following type I and II error probabilities[4]:

$$\alpha = \Pr\{T(\tau_1, \tau_2) = 0 | a_1 = a_2\}, \beta = \Pr\{T(\tau_1, \tau_2) = 1 | a_1 \neq a_2\}.$$

2.4 Ternary Voting: Indirect Comparison of Traces Using Profiling

As already mentioned in Subsection 2.1, collision detection can be made more efficient if profiling is allowed. One approach to such template-based collision detection is the ternary voting proposed in [1]: Test T on two *target traces* τ_1 and τ_2 for inputs a_1 and a_2 can be amplified by using further N *reference traces* $\{\pi_i\}_{i=1}^N$, $\pi_i = (\pi_{i,1}, \ldots, \pi_{i,l}) \in \mathbb{R}^l$, which correspond to some random unknown inputs $b_i \in \mathrm{GF}(2^8)$ and have been acquired on a similar implementation prior to the online stage. The main idea of ternary voting is to indirectly compare τ_1 and τ_2 through the pool of reference traces $\{\pi_i\}_{i=1}^N$. The ternary voting test can be defined as follows:

$$V(\tau_1, \tau_2) = \begin{cases} 0 \text{ (no collision)}, & \text{if } G(\tau_1, \tau_2) < U \\ 1 \text{ (collision)}, & \text{if } G(\tau_1, \tau_2) \geq U, \end{cases}$$

where $G(\tau_1, \tau_2) = \sum_{i=1}^N F(\tau_1, \tau_2, \pi_i)$ with $F(\tau_1, \tau_2, \pi_i) = T(\tau_1, \pi_i) \cdot T(\tau_2, \pi_i)$, U is some decision threshold, and T is the binary comparison test as defined in Subsection 2.3. The key observation is that the distributions of $G(\tau_1, \tau_2)$ for $a_1 = a_2$ and for $a_1 \neq a_2$ will be different. Typically, for sufficiently large N's $G(\tau_1, \tau_2)$ will be higher for $a_1 = a_2$ than for $a_1 \neq a_2$. To decide if there has been a collision, the attacker needs to statistically distinguish between these two distributions. We use ternary voting to amplify direct binary comparison, combining the sets of collisions detected by both methods, see Section 5.2.

3 Algebraic Collision-Based Key Recovery

In this section we identify (Subsection 3.1) types of nonlinear generalized collisions enabling efficient algebraic representation that give rise to efficient key

[4] Note that α and β strongly depend on the statistical properties of the traces (among many other factors, on the noise amplitude) and the choice of W.

recovery. Then the corresponding systems of nonlinear equations are constructed (Subsection 3.2) and solved (Subsection 3.3). We first assume that all collisions are detected correctly. We deal with collision detection errors in Subsection 3.4 and Section 4.

3.1 Nonlinear Collisions

FS-Collisions. Generalized collisions in the first two AES rounds occurring between bytes of the first two rounds are called *FS-collisions*. If input bytes $a_{j_1}^{i_1}$ and $a_{j_2}^{i_2}$ of two S-boxes collide, one has the simple linear equation over $GF(2^8)$:

$$a_{j_1}^{i_1} \oplus a_{j_2}^{i_2} = 0.$$

If a_j^i lies in the S-box layer of the first round, then $\alpha_j^i = k_j \oplus p_j^i$, for some i, j. Otherwise, one has

$$a_j^i = x_j \oplus m_{(j-1)\ div\ 4} \cdot b^i_{(j-1)\ mod\ 4+1} \oplus m_{j\ div\ 4} \cdot b^i_{j\ mod\ 4+5} \oplus$$
$$m_{(j+1)\ div\ 4} \cdot b^i_{(j+1)\ mod\ 4+9} \oplus m_{(j+2)\ div\ 4} \cdot b^i_{(j+2)\ mod\ 4+13},$$

where $m = (m_0, m_1, m_2, m_3) = (02, 03, 01, 01)$[5] and $b_j^i = S(k_j \oplus p_j^i)$.

We distinguish between the following three types of FS-collisions: linear collisions in the first round, nonlinear collisions between the first two rounds, and nonlinear collisions within the second round. These three collision types are illustrated in Figure 2. Collision 1 occurs between two bytes of the first round, linearly binding k_1 and k_{13}. Collision 2 occurs between the S-box number 7 of the second round and the S-box number 1 of the first round. It binds 6 key bytes: k_1, k_3, k_8, k_9, k_{14}, and k_7. Collision 3 algebraically connects two MixColumn expressions on 8 key bytes after the S-box layer with two bytes of the second subkey in a linear manner. The algebraic expressions in this example are the following:

$$1: \ k_1 \oplus p_1^1 = k_{13} \oplus p_{13}^1$$
$$2: \ k_1 \oplus p_1^2 = S^{-1}(s_7^2) =$$
$$x_7 \oplus 01 \cdot S(k_2 \oplus p_2^2) \oplus 02 \cdot S(k_8 \oplus p_8^2) \oplus 03 \cdot S(k_9 \oplus p_9^2) \oplus 01 \cdot S(k_{14} \oplus p_{14}^2)$$
$$3: \ s_7^3 = s_{16}^3,$$
$$k_7 \oplus 01 \cdot S(k_3 \oplus p_3^3) \oplus 02 \cdot S(k_8 \oplus p_8^3) \oplus 03 \cdot S(k_9 \oplus p_9^3) \oplus 01 \cdot S(k_{14} \oplus p_{14}^3) =$$
$$x_{16} \oplus 03 \cdot S(k_4 \oplus p_4^3) \oplus 01 \cdot S(k_5 \oplus p_5^3) \oplus 01 \cdot S(k_{10} \oplus p_{10}^3) \oplus 02 \cdot S(k_{15} \oplus p_{15}^3)$$

Note that there are also mirrored collisions occurring between the S-boxes of the last round (number 10) and the round next to the last one (number 9). Such collisions are called *LN-collisions*.

[5] Here and below any byte $uv = u \cdot 16 + v = \sum_{i=0}^{7} d_i \cdot 2^i$ is interpreted as the element of $GF(2^8) = GF(2)[\omega]$ using a polynomial representation $\sum_{i=0}^{7} d_i \cdot \omega^i$, where $\omega^8 + \omega^4 + \omega^3 + \omega + 1 = 0$ holds.

FL-Collisions. *FL-collisions* are generalized collisions between bytes of the first and last rounds. If plaintexts as well as ciphertexts are known, an FL-collision leads to a simple nonlinear equation. Linear collisions within the first round as well as those within the last round can be additionally used.

Figure 3 illustrates these three types of collisions. Collision 1 occurs between the 2nd byte of the first round and the 5th byte of the last round for some input and output with p_2^1 and c_8^1 (note that the bytes do not have to belong to the same input/output pair). Input $y_2 \oplus a^1$ to S-box 5 in the last round can be expressed as $S^{-1}(z_8 \oplus c_8^1)$ using the corresponding ciphertext and last subkey bytes. Collision 2 of Figure 3 is a linear collision within the last AES round. Collision 3 is a standard linear collision within the first AES round. The following equations result from these collisions:

$$\begin{aligned}
&1: \ k_2 \oplus p_2^1 = y_5 \oplus a^1 = S^{-1}(z_8 \oplus c_8^1), S(k_2 \oplus p_2^1) = z_8 \oplus c_8^1 \\
&2: \ y_5 \oplus a^2 = y_7 \oplus b^2, \ z_8 \oplus c_8^2 = z_6 \oplus c_6^2 \\
&3: \ k_2 \oplus p_2^3 = k_{13} \oplus p_{13}^3.
\end{aligned}$$

3.2 Constructing Systems of Equations for FS- and FL-Collisions

Equations for FS-collisions. The application of the Faugère F4 algorithm to a system of equations constructed in Subsection 3.1 for FS-collisions gives results that are superior to the linear collision attack and are summarized in Table 2.

The system of nonlinear equations is considered over GF(2). For γ inputs there are 128 variables of the first subkey K, 128 variables of the second subkey X and $128 \cdot \gamma$ intermediate variables for the output bits of the first round S-box layer. The collision-independent part of the system consists of S-box equations for the first round and linear equations for the key schedule. Since the AES S-box can be implicitly expressed as 39 equations of degree 2 [7], we have $39 \cdot 16 \cdot \gamma$ quadratic equations over GF(2) connecting the inputs and outputs of the first round S-boxes, and $4 \cdot 39 = 156$ quadratic and $12 \cdot 8 = 96$ linear equations connecting K and X using the key schedule relations. Each of the three types of FS-collisions adds 8 linear equations to the system, resulting in $8 \cdot D$ equations if D collisions occurred.

Equations for FL-collisions. FL-collisions can be also obviously expressed as a system of quadratic equations over $GF(2)$. Now we show how to derive a system of quadratic equations over $GF(2^8)$ for these collisions. One way is to use the BES expression [7]. However one would have 8 variables per one key byte in this case. We describe a simpler system, which has only 32 variables.

It is clear that linear collisions in the first or the last round can be interpreted as linear equations over $GF(2^8)$. Let us consider a nonlinear FL-collision of type 1 (see example above). Its algebraic expression is given by $S(k_{j_1} \oplus p_{j_1}^{i_1}) = z_{j_2} \oplus c_{j_2}^{i_2}$ for some $j_1, j_2 \in \{1, \ldots, 16\}$, $i_1, i_2 = 1, 2, \ldots$ Recall that the AES S-box is the composition of the multiplicative inverse in the finite field $GF(2^8)$, the $GF(2)$-linear mapping, and the XOR-addition of the constant 63. The $GF(2)$-linear

mapping is invertible, and its inverse is given by the following polynomial over $GF(2^8)$:

$$f(x) = 6e \cdot x^{2^7} + db \cdot x^{2^6} + 59 \cdot x^{2^5} + 78 \cdot x^{2^4} + 5a \cdot x^{2^3} + 7f \cdot x^{2^2} + fe \cdot x^2 + 05 \cdot x.$$

Hence we have

$$(k_{j_1} \oplus p_{j_1}^{i_1})^{-1} = f(z_{j_2}) \oplus c_{j_2}^{i_2} \oplus 63) = f(z_{j_2}) \oplus f(c_{j_2}^{i_2} \oplus 63).$$

If we replace $f(z_{j_2})$ by a new variable u_{j_2}, we obtain the quadratic equation

$$(k_{j_1} \oplus p_{j_1}^{i_1})(u_{j_2} \oplus f(c_{j_2}^{i_2} \oplus 63)) = 1,$$

which holds with probability $\frac{255}{256}$. The following proposition follows:

Proposition 1. *Solutions to the equation* $S(k_{j_1} \oplus p_{j_1}^{i_1}) = z_{j_2} \oplus c_{j_2}^{i_2}$ *coincides with solutions to the equation*

$$(k_{j_1} \oplus p_{j_1}^{i_1})(u_{j_2} \oplus f(c_{j_2}^{i_2} \oplus 63)) = 1$$

under the change of variables $u_{j_2} = f(z_{j_2})$ *with a probability of* $\frac{255}{256}$.

Moreover, if $z_{j_2} \oplus z_{j_3} = \Delta_{j_2,j_3} = c_{j_2}^{i_2} \oplus c_{j_3}^{i_3}$, then we have

$$f(z_{j_2}) \oplus f(z_{j_3}) = u_{j_2} \oplus u_{j_3} = f(\Delta_{j_2,j_3}).$$

Thus, we derive for FL-collisions the system $\mathbb{S}$ of quadratic equations over $GF(2^8)$ in 32 variables $\mathcal{K} = \{k_j, u_j\}_{1 \le j \le 16}$. Furthermore, each equation of the resulting system has only two variables. We call such equations *binomial*.

We say that a subset of variables $\mathcal{K}' \subset \mathcal{K}$ is connected, if for any non-trivial partition of $\mathcal{K}' = A \cup B$ there is an equation in $\mathbb{S}$ in two variable $v \in A$ and $w \in B$. Thus $\mathcal{K}$ can be devided into disjoint subsets $\mathcal{K}_i$ with respect to $\mathbb{S}$, where $\mathcal{K}_i$ is either connected or singleton. Each $\mathcal{K}_i$ corresponds to an unique subsystem $\mathbb{S}_i$, and we call $(\mathcal{K}_i, \mathbb{S}_i)$ a chain.

3.3 Solving Systems for FS- and FL-Collisions

Solving equations for FS-collisions. The system of nonlinear equations for FS-collisions is solved in the following way. First the system is passed to the F4 algorithm without modifications. If it is not solvable, one guesses the largest connected linear component as in linear collision-based recovery (that is, 8 bits per connected component, see Subsection 2.2), adds the corresponding linear equation to the system and tries to solve the system again. The memory limit for the Magma program was set to 500 MB. It can be seen from Table 2 that for 5 inputs most ($> 93\%$) instances of the FS-system can be solved within several hours on a PC. For 4 inputs, less systems are solvable (about 40%) within approx. 2 hours on a standard PC under Linux.

Table 2. Solving equation systems for FS-collisions over $GF(2)$

Inputs, γ	5	5	4	4
Success probability π	0.425	0.932	0.042	0.397
Offline complexity (time), s	142.8	7235.8	71.5	6456.0
Memory limit, MB	500	500	500	500
Number of variables	896	896	768	768
Linear/quadratic equations	96+8D/3276	96+8D/3276	96+8D/2652	96+8D/2652

Solving equations for FL-collisions. FL-collisions lead, as a rule, to better results. Each equation binds only two $GF(2^8)$-variables, since one deals with binomial equations introduced in Subsection 3.2 for FL-collisions. There are 32 variables $\mathcal{K}$ over $GF(2^8)$. The algebraic relations on these variables are much simpler, since one has both plaintext and ciphertext bytes (more information related to the detected collisions).

Moreover, for the system we have a set of independent chains. Let $(\mathcal{K}_i, \mathbb{S}_i)$ be a chain, and $v \in \mathcal{K}'$. Since $\mathcal{K}'$ is connected, there exists a relation between v and any other variable of $\mathcal{K}'$. It is not hard to prove that this relation can be expressed as linear or quadratic equation in two variables.Further, some chain can have a non-linear equation such that the corresponding variables still be connected also without this equation. In this case we call this chain a cycle. For any cycle the system has at most two solutions. Thus, there are often nonlinear subsystems solvable independently (see the example below).

On average, there are about 1.02 independently solvable subsystems covering 30.08 out of 32 $GF(2^8)$-variables for $\gamma = 5$ inputs and 0.99 cycles covering 20.08 out of 32 $GF(2^8)$-variables for $\gamma = 4$ inputs. Statistically there are 43.58 collisions for $\gamma = 5$ inputs and 29.66 collisions for $\gamma = 4$ inputs.

Table 3. Solving equation systems over $GF(2^8)$ for FL-collisions

Inputs, γ	5	4
Success probability π	1.00	0.85
Offline complexity (operations)	$\leq 2^{40}$	$\leq 2^{40}$
Memory limit, MB	500	500
Number of variables	32	32
Average number of equations	43.58	29.66

Table 3 contains the results for applying the F4 algorithm to FL-systems of nonlinear equations averaged over 10000 samples. After resolving the non-linear subsystems using F4, as for FS-collisions, we guess variables defining the remaining bytes in a way similar to the linear key-recovery. With $C_{\text{offline}} \leq 2^{40}$, practically all FL-systems are solvable for 5 inputs, an FL-system being solvable with probability 0.85 for $\gamma = 4$ inputs.

3.4 Key Recovery Robust to Type I Collision Detection Errors

Both algebraic and linear collision-based key recovery methods can be made tolerant to non-zero type I error probabilities of collision detection: A non-zero value of the type I collision detection error probability α is equivalent to omitting some collisions. If the number of inputs γ is somewhat increased, this is easily tolerated by the methods, since the number of true (apriori) collisions grows quadratically with the increase of the number of inputs.

Under the assumption that the type II error probability β is negligibly low, we performed this trade-off between α, γ and success probability P for FS-, FL- and linear collisions. The results can be found in Figure 4 and show that the FL-collision based method is superior to FS- and linear collision based methods even in the presence of strong noise. For example, while type I error probabilities up to $\alpha = 0.65$ are well tolerated with only $\gamma = 7$ inputs for FL-collisions, one needs at least $\gamma = 8$ or $\gamma = 9$ inputs to achieve a comparable tolerance level for FS- and linear collisions, respectively.

4 Towards Reliable Collision Detection in Practice

Probability distribution of Euclidean distance. Given two traces $\tau_1 = (\tau_{1,1}, \ldots, \tau_{1,l}) \in \mathbb{R}^l$ and $\tau_2 = (\tau_{2,1}, \ldots, \tau_{2,l}) \in \mathbb{R}^l$, we assume that each point $\tau_{i,j}$ can be statistically described as $\tau_{i,j} = s_{i,j} + r_{i,j}$, where $s_{i,j}$ is signal constant (without noise) for the given time point i as well as some fixed input to the S-box, and $r_{i,j}$ is Gaussian noise due to univariate normal distribution[6] with mean 0 and some variance σ^2 remaining the same for all time instances in our rather rough model. Let τ_1 and τ_2 correspond to some S-box inputs a_1 and a_2.

If $a_1 = a_2$, the corresponding deterministic signals are equal (that is, $s_{1,j} = s_{2,j}$ for all j's) and one has:

$$1/H(\tau_1, \tau_2)_{a_1=a_2} = \sum_{j=1}^{l} (\tau_{1,j} - \tau_{2,j})^2 = \sum_{j=1}^{l} \xi_j^2 = 2\sigma^2 \sum_{j=1}^{l} \eta_j^2,$$

where $\xi_j = r_{1,j} - r_{2,j}$, $\xi_j \sim \mathcal{N}\left(0, 2\sigma^2\right)$ and $\eta_j \sim \mathcal{N}(0,1)$. That is, statistic $1/H(\tau_1, \tau_2)_{a_1=a_2}$ follows the chi-square distribution with l degrees of freedom up to the coefficient $2\sigma^2$. As the chi-square distribution is approximated by normal distribution for high degrees of freedom, one has the following

Proposition 2. *Statistic* $1/H(\tau_1, \tau_2)_{a_1=a_2} = \sum_{j=1}^{l} (\tau_{1,j} - \tau_{2,j})^2$ *for* $\tau_i = (\tau_{i,1}, \ldots, \tau_{i,l}) \in \mathbb{R}^l$ *with* $\tau_{i,j} \sim \mathcal{N}\left(s_{i,j}, \sigma^2\right)$ *can be approximated by normal distribution* $\mathcal{N}(2\sigma^2 l, 8\sigma^4 l)$ *for sufficiently large* l*'s.*

[6] The real measured power consumption is often due to the generic multivariate normal distribution. However, almost all entries of the corresponding covariance matrix are close to zero. Thus, the model with independent multivariate normal distribution seems to be quite realistic.

Alternatively, if $a_1 \neq a_2$, one has

$$1/H(\tau_1,\tau_2)_{a_1\neq a_2} = \sum_{j=1}^{l} (\tau_{1,j}-\tau_{2,j})^2 = \sum_{j=1}^{l} \left(\delta_j^{(1,2)}+\xi_j\right)^2 = 2\sigma^2 \sum_{j=1}^{l} \nu_j^2,$$

where $\delta_j^{(1,2)} = \tau_{1,j} - \tau_{2,j}$, $\xi_j = r_{1,j} - r_{2,j}$, $\xi_j \sim \mathcal{N}\left(0, 2\sigma^2\right)$ and $\nu_j \sim \mathcal{N}\left(\delta_j^{(1,2)}/\sqrt{2}\sigma, 1\right)$. That is, statistic $1/H(\tau_1,\tau_2)_{a_1\neq a_2}$ follows the *noncentral* chi-square distribution with l degrees of freedom and $\lambda = \sum_{j=1}^{l} \left(\delta_j^{(1,2)}/\sqrt{2}\sigma\right)^2$ up to the coefficient $2\sigma^2$. Again, we have an approximation using

Proposition 3. *Statistic* $1/H(\tau_1,\tau_2)_{a_1\neq a_2} = \sum_{j=1}^{l} (\tau_{1,j}-\tau_{2,j})^2$ *for* $\tau_i = (\tau_{i,1},\ldots,\tau_{i,l}) \in \mathbb{R}^l$ *with* $\tau_{i,j} \sim \mathcal{N}\left(s_{i,j},\sigma^2\right)$ *can be approximated by normal distribution* $\mathcal{N}\left(2\sigma^2(l+\lambda), 8\sigma^4(l+2\lambda)\right)$ *with* $\lambda = \sum_{j=1}^{l} \left(\delta_j^{(1,2)}/\sqrt{2}\sigma\right)^2$ *for sufficiently large* l*'s.*

Selection of most informative trace points. In the direct binary comparison, we try to distinguish between the distributions $1/H(\tau_1,\tau_2)_{a_1\neq a_2}$ and $1/H(\tau_1,\tau_2)_{a_1=a_2}$. As described above these statistics approximately follow normal distribution for large numbers of trace points. That is, to efficiently distinguish between these two statistics it is crucial to decrease their variances while keeping the difference of their means high. For this purpose, to increase the success probability of the Euclidean distance test, we propose to discard points of traces with small minimal contribution to the difference of means.

To illustrate this method of point selection, we assume for the moment that $\delta_j^{(1,2)} = 0$ for $j > l/2$ and $\delta_j^{(1,2)} \neq 0$ for $j \leq l/2$ with l even, that is, the second half of the trace does not contain any data dependent information. Then we can discard the second halves of the both traces τ_1 and τ_2 in the direct binary comparison function and compute two related statistics on the rest of the points:

$$1/H'(\tau_1,\tau_2)_{a_1=a_2} = \sum_{j=1}^{l/2} (\tau_{1,j}-\tau_{2,j})^2, 1/H'(\tau_1,\tau_2)_{a_1\neq a_2} = \sum_{j=1}^{l/2} (\tau_{1,j}-\tau_{2,j})^2.$$

This will adjust the means and variances of the approximating normal distributions: $\mathcal{N}\left(\sigma^2 l, 4\sigma^4 l\right)$ and $\mathcal{N}\left(2\sigma^2(l/2+\lambda), 8\sigma^4(l/2+2\lambda)\right)$, respectively. Note that the difference of means remains unaffected and equal to $2\sigma^2\lambda$. At the same time both variances are reduced, one of them by factor 2, which allows one to distinguish between these two distributions more efficiently and, thus, to detect collisions more reliably.

More generally speaking, for AES we have to reliably distinguish between inputs in each (a_{i_1}, a_{i_2}) of the $\binom{256}{2}$ pairs of byte values, $a_{i_1}, a_{i_2} \in \mathrm{GF}(2^8)$. Thus, the most informative points j of the traces are those with maximal minimums of $\delta_j^{(i_1,i_2)}$ over all pairs of different inputs, that is, points j with maximal

values of

$$\min_{a_{i_1} \neq a_{i_2}} \delta_j^{(i_1,i_2)}.$$

We estimated these values for all time instances j of our AES implementation and compared this to the signal variance in the same time points, $\mathrm{var}(s_{i,j})$, $a_i \in \mathrm{GF}(2^8)$, which is known to be a good indicator of the points leaking information in DPA. This comparison is represented in Figure 5 for two clock cycles of our 8-bit table look-up operation.

5 Experimental Validation

5.1 AES Implementation and Measurement Equipment

We performed our attacks for a typical AES implementation on the Atmel ATmega16 microcontroller, an RISC microcontroller from the 8-bit AVR family with a Harvard architecture. 8-bit AVR microcontrollers are widely used in embedded devices. To run a collision attack, the attacker has to know when the AES S-boxes are executed. So we measured the power consumption of the table look-ups corresponding to the relevant S-box applications. These include instances in SubBytes, ShiftRows, and MixColumns operations.

The microcontroller was clocked at 3.68 MHz and supplied with an operating voltage of 5V from a standard laboratory power source. The variations of the power consumption were observed on a shunt resistor of 5.6 Ohm inserted into the ground line of the microcontroller. The measurements were performed with a LeCroy WaveRunner 104MXi DSO equipped with ZS1000 active probe. The DSO has 8-bit resolution and 1 GHz input bandwidth (with the specified probe). The acquisitions were performed at the maximum sampling rate of 10 GS/s without any input filters. We stress that this measurement setup is *not* noise-optimized[7].

5.2 Attack Scenarios and Results

Performance metric. We use the following efficiency metric to compare the performance of all these attacks: $t \cdot \gamma / P$, where t is the number of averagings, γ is the number of different inputs, and P is the success probability of the attack. In case of the Hamming-distance based CPA we apply a similar metric: n/p^{16}, where n is the number of measurements needed to determine a single-byte chunk of the

[7] As in case of DPA [8], collision detection methods tend to be sensitive to the pre-processing of measured signals. To denoise the traces, we proceed in two steps. First, the traces are decimated by applying a low-pass filter to the original traces and subsequently resampling them at a lower rate. Additionally to noise reduction, this weakens time jitter. Second, the decimated traces are denoised by applying a wavelet decomposition at a certain level, thresholding the detail coefficients, and subsequent wavelet reconstruction. Our experiments show that symlets proposed by Daubechies (first of all, the 'sym3' wavelet) are most suitable for this operation.

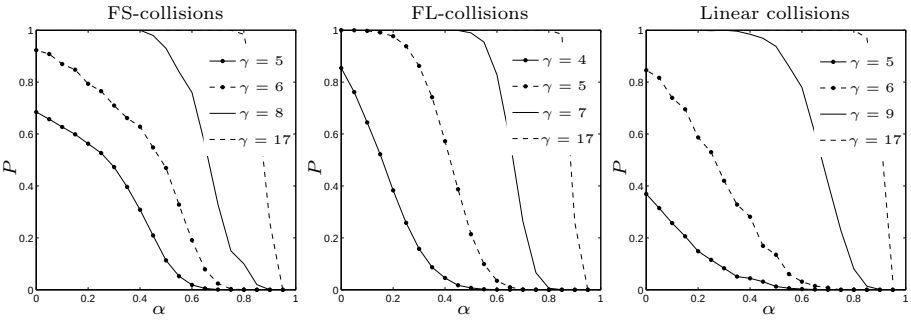

Fig. 4. Success probability P against type I error probability α in FS-, FL- and linear collision-based key recovery for different numbers γ of random inputs

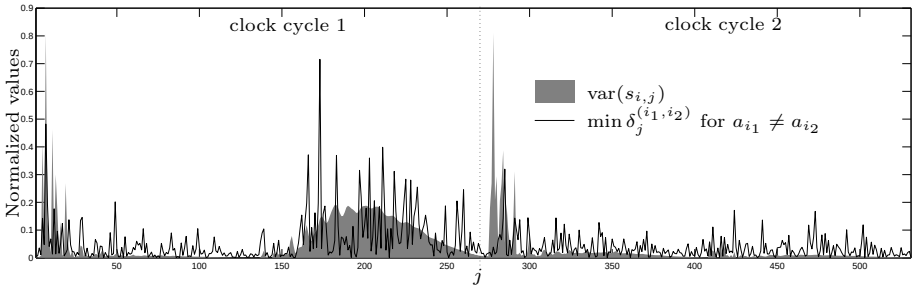

Fig. 5. Informative points for collision detection and DPA

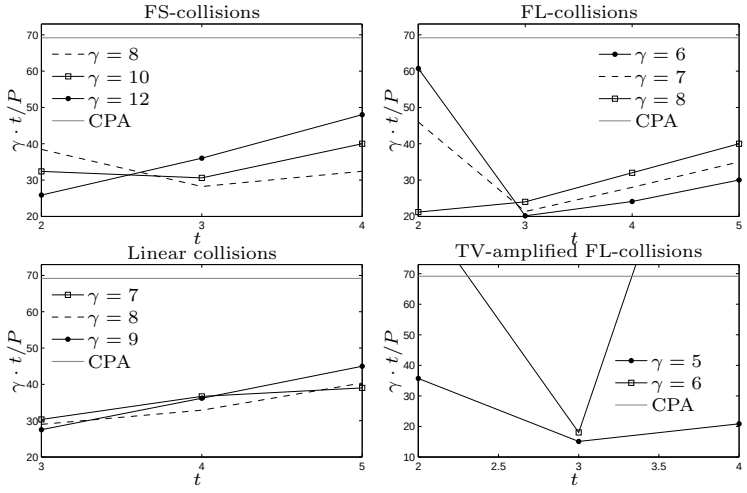

Fig. 6. Performance of collision attacks based on FS-, FL- and linear collisions without profiling as well as of FL-collision based attacks with profiling (ternary voting with 625 profiling measurements) vs. Hamming-distance based CPA on the same traces

AES key with probability p. These metrics characterize the expected number of measurements needed to recover the whole 16-byte key. The performance results for CPA can be found in Figure 6.

Collision attacks without profiling. In the online stage, an attacker observes executions of AES for γ random inputs, each repeated t times. Then, the $t \cdot \gamma$ traces are averaged t times and one obtains γ averaged traces for γ random inputs. These are used to detect collisions – linear, FS- or FL-collisions (see Sections 2 and 3) depending on the key-recovery method – with the direct binary comparison (see Subsection 2.3). All key-recovery methods need AES plaintexts to be known. Additionally, if the attack is based on FL-collisions, the corresponding AES ciphertexts have to be known.

Note that since the adaptive threshold W is used in the direct binary comparison which eliminates all type II errors, many true collisions are omitted due to the increased type I error probability by shifting W to the right. That is, for given γ and α, the success probability P of the whole attack follows the dependencies illustrated in Figure 4 for different key-recovery methods.

Profiling-amplified collision attacks. If profiling is possible prior to the online stage, the ternary voting method (see Subsection 2.4) can be used to detect additional collisions, which are omitted by the direct binary comparison due to the application of the adaptive threshold. Taking additional collisions is equivalent to the increase of α, which further improves the performance metric $t \cdot \gamma / P$. In our profiling-amplified attacks we used $N = 10^5$ profiling S-box traces τ_i which is equivalent to about $10^5/160 = 625$ executions of AES in the profiling stage. See Figure 6 for concrete results of collision attacks with profiling.

References

1. Bogdanov, A.: Multiple-differential side-channel collision attacks on AES. In Oswald, E., Rohatgi, P., eds.: CHES'08. Volume 5154 of LNCS., Springer-Verlag (2008) 30–44
2. Kocher, P.C., Jaffe, J., Jun, B.: Differential power analysis. In Wiener, M., ed.: CRYPTO'99. Volume 1666 of LNCS., Springer-Verlag (1999) 388–397
3. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In Kaliski Jr., B.S., Koç, Ç.K., Paar, C., eds.: CHES'02. Volume 2523 of LNCS., Springer-Verlag (2003) 51–62
4. Lemke-Rust, K.: Models and Algorithms for Physical Cryptanalysis. PhD thesis, Ruhr University Bochum (2007)
5. Schramm, K., Leander, G., Felke, P., Paar, C.: A collision-attack on AES: Combining side channel- and differential-attack. In Joye, M., Quisquater, J.J., eds.: CHES'04. Volume 3156 of LNCS., Springer-Verlag (2004) 163–175
6. Bogdanov, A.: Improved side-channel collision attacks on AES. In Adams, C., Miri, A., Wiener, M., eds.: SAC'07. Volume 4876 of LNCS., Springer-Verlag (2007) 84–95
7. Cid, C., Murphy, S., Robshaw, M.: Algebraic Aspects of the Advanced Encryption Standard. Springer-Verlag (2006)
8. Charvet, X., Pelletier, H.: Improving the DPA attack using wavelet transform. NIST Physical Security Testing Workshop (2005)