Distinguisher and Related-Key Attack on the Full AES-256 (Extended Version)

Alex Biryukov, Dmitry Khovratovich, Ivica Nikolić

University of Luxembourg

{alex.biryukov, dmitry.khovratovich, ivica.nikolic@uni.lu}

Abstract. In this paper we construct a chosen-key distinguisher and a related-key attack on the full 256-bit key AES. We define a notion of *differential q-multicollision* and show that for AES-256 q-multicollisions can be constructed in time $q \cdot 2^{67}$ and with negligible memory, while we prove that the same task for an ideal cipher of the same block size would require at least $O(q \cdot 2^{\frac{q-1}{q+1}128})$ time. Using similar approach and with the same complexity we can also construct q-pseudo collisions for AES-256 in Davies-Meyer hashing mode, a scheme which is provably secure in the ideal-cipher model. We have also computed partial q-multicollisions in time $q \cdot 2^{37}$ on a PC to verify our results. These results show that AES-256 can not model an ideal cipher in theoretical constructions. Finally we extend our results to find the first publicly known attack on the full 14-round AES-256: a related-key distinguisher which works for one out of every 2^{35} keys with 2^{120} data and time complexity and negligible memory. This distinguisher is translated into a key-recovery attack with total complexity of 2^{131} time and 2^{65} memory.

Keywords: AES, related-key attack, chosen key distinguisher, Davies-Meyer, ideal cipher.

1 Introduction

The Advanced Encryption Standard (AES) is a block cipher which was chosen by NIST from a set of 15 candidate designs in a thorough evaluation process that lasted from September 1997 till October 2000. On November 26, 2001 Rijndael [6], a 128-bit block, 128/192/256-bit key block cipher has become a standard as U.S. FIPS 197 [13]. In June 2003 the US government has approved the use of 128, 192, 256 bit key AES for SECRET and 192, 256-bit key AES for TOP SECRET information [14]. In the last ten years AES has been subject to very intensive cryptanalytic effort, with best currently known attacks breaking 7, 10, 10 rounds for respective keysizes (128, 192, 256), with very high complexities.

In this paper we show for the first time in the open literature distinguishers and related-key attacks on the full 14-round 256-bit key AES. Research presented in this paper follows the logic described in Fig. 1. First we identified slow diffusion and other differential weaknesses in the key schedule of AES-256 which match nicely with the differential properties of the round function. This allows us to construct local collisions for AES, i.e. two round difference propagation patterns which result in low weight difference in the subkeys and zero difference in the 128-bit block. We concatenate four such local collisions together and add another 6-round trail on top in order to cover full 14 rounds of AES-256. The trail[1] has 41 active S-boxes (36 in the block and 5 in the key schedule), so we apply a special tool, a triangulation algorithm (designed for the purpose of finding collisions in hash functions), in order to find keys and plaintexts that conform to the trail.

From this point we go in two directions. First we show that for AES-256 one can construct a *chosen-key* distinguisher based on the new notion of a *differential q-multicollision* in time $q \cdot 2^{67}$ and with negligible memory. We prove that the same task for an ideal cipher of the same block size would require at least $q \cdot 2^{\frac{q-1}{q+1}128}$ time for $q \leq 57$ and at least $q \cdot 2^{\frac{q-2}{q+2}128}$ for $q > 57$. I.e. for

[1] We use colors in the diagrams of the trails, so please refer also to the tables in the appendix if you print this paper on a black and white printer.

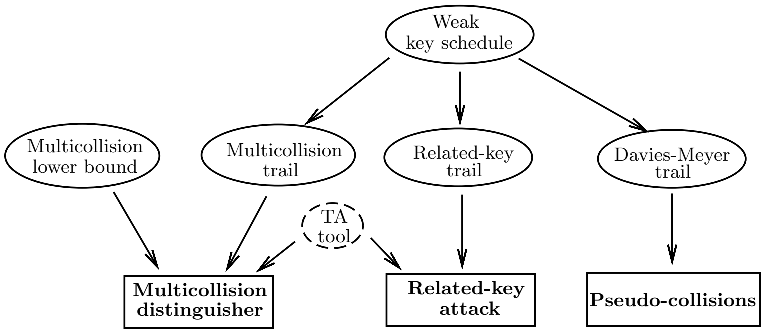

Fig. 1. Outline of the research presented in this paper.

$q > 3$ the differential multicollision for AES-256 can be constructed significantly faster than for an ideal cipher. Previously a *known-key* distinguisher for seven rounds of AES with 2^{56} texts was found in [12]. To verify our results we found partial q-multicolisions in several hours on a PC using the publicly available implementation of AES-256.

As a direct application of this differential q-multicollision distinguisher we show that AES-256 when used in the Davies-Meyer mode allows to construct q pseudo-collisions with *fixed* differences ΔIV, ΔM in the IV and the message with complexity $q \cdot 2^{67}$. Again, such a result would require at least $q \cdot 2^{\frac{q-2}{q+2}128}$ time for the ideal cipher in the Davies-Meyer mode. Results of this type try to enchance our definitions of block cipher security and to fill the gap between theoretical models like random oracle and ideal cipher and the real world of ciphers which have fixed description and are efficiently computable [2,5]. However a proper security definition which would capture the intuition behind chosen/known key attacks is still an open problem.

The second direction that we studied was application of the trails that we have found to more standard attacks on a block cipher, for example related-key attacks. In particular we show that by changing the top two rounds of the trail that we used previously one obtains a differential trail with only 24 active S-boxes (19 in the round function and 5 in the key schedule). From this trail we can construct a differential distinguisher for AES-256 which works for one key out of 2^{35} and has complexity 2^{120} data and time, and negligible memory. This distinguisher can be used to mount a key-recovery attack on AES-256 with total complexity of $2^{35+96} = 2^{131}$ time and 2^{65} memory. We summarize our findings in Table 1.

This paper is organized as follows: in Section 2 we prove a lower bound on the complexity of finding differential q-multicollisions in the case of an ideal cipher and construct a distinguisher for the full AES-256. In Section 3 we show an application of these results to finding pseudo-collisions for the Davies-Meyer hashing mode instantiated with AES-256. In Section 4 we show a related-key attack on the full AES-256. In Sections 5 and 6 we discuss the relevance of our attack and new design criteria for block cipher key schedule as a consequence of our attack. Section 7 concludes the paper. In Appendix we provide technical details about our differential trails and the triangulation algorithm.

Discussion. It is clear that the *open key* (chosen or known) security model is new and is still lacking a proper security definition. However we think that if we can support an open-key attack with a proof of security against such attack for the ideal cipher, this gives us additional confidence that such property (in our case "differential multicollisions") should not be present in a good cipher. There are also many constructions provably secure [3,8] in the ideal cipher model. This model assumes that both the key and the plaintext are accessible to the attacker. If a block cipher (e.g., AES) exhibits a property that should not appear in the ideal cipher then instantiation of a provably secure construction with this cipher could undesirably weaken the construction. Our

Attack	**# rounds**	**# keys**	**Data**	**Time**	**Memory**	**Source**
Known-key integral	7	1	2^{56}	2^{56}	2^{56}	[12]
Partial sums	9	256	2^{85}	2^{226}	2^{32}	[7]
Related-key rectangle	10	64	2^{114}	2^{173}	?	[1,11]
q-multicollisions	14	$2q$	$2q$	$q \cdot 2^{67}$	-	Sec. 2
Partial q-multicollisions	14	$2q$	$2q$	$q \cdot 2^{37}$	-	Sec. 2.3
Related-key distinguisher	14	2^{35}	2^{119} *	2^{119} *	-	Sec. 4.1
Related-key key recovery	14	2^{35}	2^{96} *	2^{96} *	2^{65}	Sec. 4.2

* — for each key.

Table 1. Best attacks on AES-256

Davies-Meyer example is exactly to show that a construction provably secure in the ICM can break down if instantiated with AES-256 (such a hash function was never proposed for another reason — 128-bit state is too short for a modern hash). The fact that this property does not automatically carry on to Davies-Meyer instantiated with all the other block-ciphers (hopefully), shows a non-trivial weakness of AES-256.

2 Multicollision distinguisher

In this section we provide a chosen-key distinguisher for AES-256 which has practical complexity.

Definition 1. *A set of two differences and q pairs*

$$\{\Delta_K, \Delta_P;\ (P_1, K_1),\ (P_2, K_2), \ldots, (P_q, K_q))\}$$

is called a differential q-multicollision *for a cipher $E_K(\cdot)$ if*

$$E_{K_1}(P_1) \oplus E_{K_1 \oplus \Delta_K}(P_1 \oplus \Delta_P) = E_{K_2}(P_2) \oplus E_{K_2 \oplus \Delta_K}(P_2 \oplus \Delta_P) = \\ = \cdots = E_{K_q}(P_q) \oplus E_{K_q \oplus \Delta_K}(P_q \oplus \Delta_P). \quad (1)$$

A differential q-multicollision can be also viewed as a set of q right pairs with respect to the related-key differential, where the key is not fixed.

We compare the task of constructing a differential q-multicollision for an ideal cipher with that for AES-256. This task for an ideal cipher, i.e. a set of 2^k randomly chosen permutations, would require treating it as a black-box and making only encryption/decryption queries. We expect that for a good cipher with no (yet discovered) structural flaws, the task of constructing a differential q-multicollision would have the same complexity as for an ideal cipher.

Let us compute this complexity measured in the number of queries. Since the cipher is ideal, an adversary is only given an access to the encryption and decryption oracles, both having two inputs (a key and a plaintext/ciphertext) and one output. This is the same model of an adversary as in [2,3, p. 329].

In the beginning no triplet ⟨*plaintext*, *key*, *ciphertext*⟩ is defined. Then, for each query of the adversary "$E_K(P) =?$" the encryption oracle takes a random value C from a possible range (where $E_K(\cdot)$ is yet undefined) and thus defines $E_K(P) = C$. Also $E_K^{-1}(C)$ becomes defined. The same rule holds for a decryption query.

Lemma 1. *To construct a differential q-multicollision for an ideal cipher with an n-bit block an adversary needs at least $O(q \cdot 2^{\frac{q-2}{q+2}n})$ queries on the average.*

Proof. See Sec. 2.1.

Remark 1. For small q, when the lower bound does not exceed 2^{n-1}, a better estimate is obtained (see the proof of the lemma). In our case, for $n = 128$, an adversary needs at least $q \cdot 2^{\frac{q-1}{q+1}128}$ queries if $q \leq 57$.

Surprisingly, differential multicollisions for AES-256 can be constructed substantially faster. Furthermore, we can set $\Delta_P = 0$ in a multicollision, so a stronger statement holds.

Theorem 1. *A differential q-multicollision with $\Delta_P = 0$ for AES-256 can be found with time complexity $q \cdot 2^{67}$.*

Proof. See Sec. 2.2.

Thus for $q > 3$ a differential q-multicollision for AES-256 can be constructed significantly faster than for an ideal cipher.[2] Therefore, AES-256 can not model an ideal cipher.

2.1 Proof of Lemma 1

Proof. Let A be an adversary attacking the cipher, and assume that A asks its oracles a total of L queries, where $L < 2^{n-1}$. Assume that a multicollision of the form (1) is found. Let us compute the probability of this event. First, we rewrite (1) as $U_1 = U_2 = \cdots = U_q$. With each term $U_j = E_{K_j}(P_j) \oplus E_{K_j \oplus \Delta_K}(P_j \oplus \Delta_P)$ we associate an integer t_j such that t_j-th oracle query determines the value of U_j, i.e., computes the last (chronologically) element of the sum. Without loss of generality, assume that $t_1 < t_2 < \cdots < t_q$. Finally, define t'_1 as the index of the query that determines the first element of the sum U_1.

$$\overbrace{\underbrace{E_{K_1}(P_1)}_{\text{queried at } t'_1} \oplus \underbrace{E_{K_1 \oplus \Delta_K}(P_1 \oplus \Delta_P)}_{\text{queried at } t_1}}^{U_1} = \overbrace{\underbrace{E_{K_2}(P_2)}_{\text{queried before } t_2} \oplus \underbrace{E_{K_2 \oplus \Delta_K}(P_2 \oplus \Delta_P)}_{\text{queried at } t_2}}^{U_2} =$$

$$= \cdots = \overbrace{\underbrace{E_{K_q}(P_q)}_{\text{queried before } t_q} \oplus \underbrace{E_{K_q \oplus \Delta_K}(P_q \oplus \Delta_P)}_{\text{queried at } t_q}}^{U_q}. \quad (2)$$

Now compute for every $(t'_1, t_1, t_2, t_3, \ldots, t_q)$ the probability that this set defines a differential q-multicollision. Before submitting t_i-th query, $i > 1$, the following equation holds:

$$U_1 = U_2 = \cdots = U_{i-1},$$

where terms of $U_1, U_2, \ldots, U_{i-1}$ are completely determined by a tuple $(t'_1,\ t_1,\ t_2,\ t_3,\ \ldots,\ t_{i-1})$. Indeed, from t'_1 and t_1 we define K_1, Δ_K, P_1, Δ_P; from t_j we define K_j and P_j.

Just before the moment t_i only one term of U_i is computed — w.l.o.g. let it be $E_{K_i}(P_i)$. Thus the equality $U_{i-1} = U_i$ should hold, i.e.

$$U_{i-1} = E_{K_i}(P_i) \oplus \underbrace{E_{K_i \oplus \Delta_K}(P_i \oplus \Delta_P)}_{\text{queried at } t_i}$$

By our definition, t_i is the first moment when $E_{K_i \oplus \Delta_K}(P_i \oplus \Delta_P)$ is queried. Then either the decryption or the encryption oracle is called. In the first case the decryption oracle is called with a ciphertext C and a key K, which for some i should be equal to $K_i \oplus \Delta_K$. By the definition of t_i, the value C is chosen from the set where $E_{K_i \oplus \Delta_K}(\cdot)$ is undefined. To become a part of a

[2] Moreover, even for $q = 3$ we are not aware of any algorithm faster than $2^{2n/3}$.

multicollision, there should exist P_i such that $C = E_{K_i}(P_i) \oplus U_{i-1}$. On the other hand, after the decryption oracle is called, the following equation should hold:

$$E^{-1}_{K_i \oplus \Delta_K}(C) = P_i \oplus \Delta_P. \tag{3}$$

Since $L < 2^{n-1}$, not more than 2^{n-1} texts were encrypted or decrypted with the key $K_i \oplus \Delta_K$. So the probability that (3) holds does not exceed $1/2^{n-1}$.

In the second case, let the encryption oracle be queried with a plaintext P and a key K, which for some i should be equal to $K_i \oplus \Delta_K$. For an answer C, a similar equation should hold:

$$C = U_{i-1} \oplus E_{K_i}(P_i). \tag{4}$$

The same probability argument holds for this equation. Therefore, for every $i \geq 2$ we get a multiplier 2^{1-n} to the probability that a tuple $(t'_1, t_1, t_2, t_3, \ldots, t_q)$ defines a differential q-multicollision. There are $\binom{L}{q+1}$ such tuples, each defining a differential q-multicollision with probability at max $2^{(q-1)(1-n)}$. We get the following equation for the number of queries required to get a q-multicollision with probability 1/2:

$$\binom{L}{q+1} \geq 2^{(q-1)(n-1)-1}. \tag{5}$$

Let us simplify the left part:

$$\binom{L}{q+1} = \frac{L!}{(L-q-1)!(q+1)!} = \frac{L(L-1)\cdots(L-q)}{(q+1)!} \leq$$
$$\leq \frac{L^{q+1}}{(q+1)!} \leq \frac{L^{q+1}}{\frac{(q+1)^{q+1}}{e^{q+1}}} = \left(\frac{eL}{q+1}\right)^{q+1}. \tag{6}$$

Substitute the result to (5):

$$\left(\frac{eL}{q+1}\right)^{q+1} \geq 2^{(q-1)(n-1)-1} \;\Rightarrow\; L \geq \frac{q+1}{e} 2^{\frac{q-1}{q+1}(n-1)-1} = O(q \cdot 2^{\frac{q-1}{q+1}n}). \tag{7}$$

This is the bound for the number of queries needed to construct a multicollision with probability 1/2. By Markov's inequality, the average number of queries exceeds this bound divided by two, so the right part of (7) is still a correct lower bound.

Now consider the case when $L \geq 2^{n-1}$. Let $\overline{K}$ be the set of keys such that there were more than 2^{n-1} encryption or decryption queries on each of these keys. Define $l = |\overline{K}|$. If $l > q-2$ then L exceeds $q \cdot 2^{n-2}$, which implies the statement of the lemma. If $l \leq q-2$ then there are at least $q-l$ sums U_i in (2) that do not involve keys from $\overline{K}$. So if a q-multicollision has been found, then a $(q-l)$-multicollision has too been found such that it does not involve keys from $\overline{K}$. Then all the arguments on the probability of this event can be carried out from the first part of the proof.

Therefore, we gets the following inequality on L:

$$L \geq \frac{q-l+1}{e} 2^{\frac{q-l-1}{q-l+1}(n-1)-1} + l \cdot 2^{n-1}.$$

For $l < q/2$ we get the following:

$$L \geq \frac{q}{2e} 2^{\frac{q-2}{q+2}(n-1)-1} = O(q \cdot 2^{\frac{q-2}{q+2}n}). \tag{8}$$

For $l \geq q/2$ we get

$$L \geq q \cdot 2^{n-2} = O(q \cdot 2^n). \tag{9}$$

Equations (7), (8), and (9) complete the proof. □

Remark 2. The function $F_{\Delta_K,\Delta_P}(K,P) = E_K(P) \oplus E_{K\oplus\Delta_K}(P \oplus \Delta_P)$ is a xor of two permutations. Patarin in [15] has shown that the xor of two random permutations can not be distinguished from a pseudo-random function with less than 2^n queries. In [16] it was proven that q-multicollision search for a random function requires at least $(q!)^{\frac{1}{q}} 2^{\frac{q-1}{q}n}$ effort. In our case we can not use this result since it assumes that Δ_K and Δ_P are fixed in advance while we allow an attacker to choose them during the attack.

2.2 Proof of Theorem 1

Here we construct a differential q-multicollision (1) with $\Delta_P = 0$ in $q \cdot 2^{67}$ time. This is done in 5 steps:

1. Build a differential trail, which is efficient for the multicollision search.
2. Derive Δ_K from the trail.
3. Choose the active S-boxes, whose inputs will be fixed in the triangulation algorithm. Denote this set by S.
4. Run the triangulation algorithm and derive a set of free variables.
5. Produce q pairs (P, K) for (1) as follows:
   (a) Assign inputs to S-boxes from S with admissible values.
   (b) Assign free variables randomly.
   (c) Produce (P, K).
   (d) Check if (P, K) and $(P, K \oplus \Delta_K)$ fit (1).

We expect that most of our readers are familiar with the description of AES and thus point out only main features of AES-256 that are crucial for our proof.

Differential trail

Notations. Differential trail (also called differential characteristic) is a sequence of differences in all the internal states of the cipher and all the subkeys. If we distinguish only between zero and non-zero (byte) differences, we call such a trail *a trail with truncated differences.*

We denote the subkey of round i by K^i, i.e. the first (whitening) subkey is K^0, the subkey of round 1 is K^1, etc., the last subkey is K^{14}. The difference in K^i is denoted by ΔK^i. Bytes of a subkey are denoted by $K^l_{i,j}$, where i stands for the row index, and j stands for the column index in the standard matrix representation of AES. Bytes of the plaintext are denoted by $P_{i,j}$, and bytes of the internal state after the SubBytes transformation in round r are denoted by $A^r_{i,j}$. Let also $B^r_{i,j}$ denote a byte in position (i, j) after the r-th application of MixColumns.

Features of AES-256. AES-256 has 14 rounds and a 256-bit key, which is two times larger than the internal state. Thus the key schedule consists of only 7 rounds. One key schedule round consists of the following transformations:

$$\begin{aligned}
K_{i,0} &\leftarrow S(K_{i+1,7}) \oplus K_{i,0} \oplus C_r, && 0 \le i \le 3;\\
K_{i,j} &\leftarrow K_{i,j-1} \oplus K_{i,j}, && 0 \le i \le 3,\ 1 \le j \le 3;\\
K_{i,4} &\leftarrow S(K_{i,3}) \oplus K_{i,4}, && 0 \le i \le 3;\\
K_{i,j} &\leftarrow K_{i,j-1} \oplus K_{i,j}, && 0 \le i \le 3,\ 5 \le j \le 7,
\end{aligned} \tag{10}$$

where $S()$ stands for S-box, and C_r — for the round-dependant constant. Therefore, every round has 8 S-boxes.

Weakness in the key schedule. Two features of the key schedule help us to build a good differential trail. First, the key schedule has a slow diffusion in backward direction. It means that a difference in a single byte $K_{0,0}$ will propagate to only two bytes, $K_{0,0}$ and $K_{0,1}$, if we apply the inversion of the key schedule round. The next inverted round will affect only one more byte, etc. Thus we can build a trail with a low-weight difference in key schedule if we start with a low-weight difference in the last round and then step backwards.

The second feature is unique to AES-256 due to its "key size"/"state size" ratio, $N_k/N_b = 2$. We can inject "good" values with the first part of the key and then cancel them, after they pass the round, with the second part of the key. We call this a *local collision* (Fig. 2).

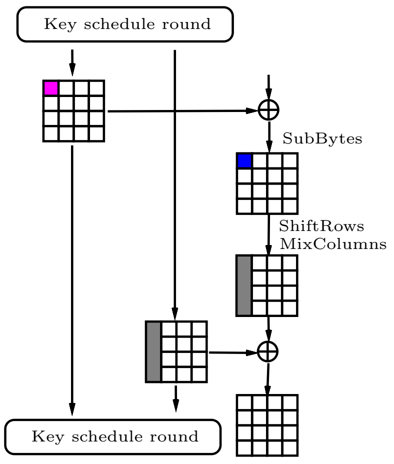

Fig. 2. A local collision.

Constructing a trail. Step by step, we construct a differential trail from the last rounds to the first ones. The trail is described in details in Appendix A, both in a truncated form and with the actual differences given. The trail has 41 active S-boxes, and 5 of them are in the key schedule.

Search for a solution. After the trail has been defined, we produce a pair (P, K) which with pair $(P, K + \Delta_K)$ fits the trail and thus is a part of a differential multicollision (1) since all such pairs have the same difference in the ciphertext. Now note that the trail explicitly states all the non-zero input δ_I and output δ_O differences of the S-boxes. According to the S-box properties, there are at most 4 solutions of the equation $S(x \oplus \delta_I) \oplus S(x) = \delta_O$. This set of solutions we call *admissible inputs*. Therefore, (P, K) and $(P, K + \Delta_K)$ fit the trail if and only if in the execution $E_K(P)$ all active S-boxes get admissible inputs. In the next paragraphs we explain how to construct such executions efficiently.

Triangulation algorithm

Search for free variables. The triangulation algorithm was proposed in [10] as a tool for solving systems of non-linear equations, which appear in differential attacks. Given the constraints on the internal variables, the algorithm outputs a special set of variables, called *free variables*. These free variables can be assigned randomly; and this assignment together with pre-fixed variables completely and efficiently determines the whole execution. The fewer variables are fixed the better the algorithm works.

Our goal is to efficiently produce the cipher executions in which active S-boxes get admissible values. However, the algorithm can not process all the active S-boxes of our trail since they are positioned too far from one another.

We found that we can fix inputs to 30 out of 41 S-boxes: all the active S-boxes in the internal states of rounds 1–4 and all the 5 active S-boxes in the key scheduling. The triangulation algorithm outputs 18 free variables, out of these 11 are in the key and provide freedom in the choice of the key for the distinguisher. These free variables are listed below.

Key	Internal state
$K^1_{2,0},\ K^1_{3,0},\ K^1_{3,1},\ K^2_{0,1},\ K^3_{0,1}$	$A^2_{0,1},\ B^2_{1,1},\ B^2_{1,2}$
$K^3_{2,0},\ K^3_{3,0},\ K^3_{2,1},\ K^4_{0,1},\ K^5_{0,1},\ K^5_{3,3}$	$B^2_{2,3},\ B^2_{3,3},\ A^3_{1,1},\ A^3_{1,2}$

Constructing a pair. Having assigned 18 free variables randomly and 30 S-box inputs with an admissible value, we substitute these values to the equations, which have been ordered by the triangulation algorithm. One by one, all the variables are determined and thus a pair (P, K) is defined. It fits the trail if and only if the 11 S-boxes not covered by the triangulation algorithm get admissible values as inputs. For the S-box in round 6 only 2 values are admissible so the probability is 2^{-7} while for the other 10 S-boxes 4 values are admissible. This results in the overall probability $2^{-(7+10\cdot 6)} = 2^{-67}$. Thus out of 2^{67} pairs one fits the trail on average. We wrote a program and checked that the distribution of the pairs is random enough so the probability estimates are likely to be correct. We also checked experimentally that bottom 7 rounds of AES produce expected difference after 2^{30} pairs on the average, exactly as predicted by the trail.

The complexity of the attack. Recall the scheme of the attack, which was given in the beginning of the proof. The first four steps are precomputations and actually have negligible cost. The triangulation algorithm works less than a second. The last step requires only to substitute the values into the equations one by one, which is computationally equivalent to a single encryption. Thus to get a right pair we need about 2^{67} operations each equivalent to one encryption. The attack needs negligible memory and is fully parallelizable.

2.3 Practical distinguisher

The definition of differential q-multicollision can be further relaxed if we allow arbitrary difference at some byte positions of Δ_P, Δ_K or Δ_C. Although an attacker gets more freedom, finding such a construction for an ideal cipher becomes easier as well. To get a lower bound, only a slight modification of Lemma 1 is required.

For the 13 rounds of AES-256 the complexity of finding this type of differential 5-multicollision, with fixed difference in 14 bytes of the plaintext and fixed ciphertext difference can be lower bounded by $2^{\frac{4\cdot 112}{6}} = 2^{74.6}$ computations. For the full AES-256 a differential 10-multicollision with half of the plaintext difference fixed, and the fixed ciphertext difference the lower bound is $2^{\frac{9\cdot 64}{11}} = 2^{52.3}$ computations. Note that these lower bounds are far from being tight. In practice we expect an efforts of 2^{112} and 2^{64} for finding *each* extra collision for 13 and 14 rounds of AES-256 respectively, since the differences are structured and *fixed*.

At the same time we can do it much faster, in just $q \cdot 2^{37}$ in both cases, which allowed us to compute these distinguishers in several hours on a PC. The actual values will be given in the extended version of this paper. The core of a practical distinguisher is a multicollision trail (Figure 5), where the behavior of S-boxes in the first two rounds is not restricted. Computing from the middle, we get 14 bytes with fixed difference before the second round, and 8 bytes with fixed difference before the first round. The triangulation algorithm covers all but six active S-boxes in rounds 3–14 so that we find a (partial) q-multicollision with complexity $q \cdot 2^{37}$.

3 Pseudo-collisions for AES-based hashing

The Davies-Meyer mode of blockcipher-based hashing has been proven collision-resistant if instantiated by an ideal cipher [3]. In this section we show a similar proof in the ideal-cipher model for the q pseudo-collision resistance, when differences in the IV and the message (Δ_I, Δ_M) are fixed. We then show that it is relatively easy to find q pseudo-collisions for AES-256 in the Davies-Meyer mode. We also point out that we construct one-block pseudo-collisions and thus the technique of Joux [9] does not apply here.

Our goal is for fixed differences Δ_I, Δ_M to find many pseudo-collisions for the $H^E(I, M) \overset{\text{def}}{=} E_M(I) \oplus I$ which is the Davies-Meyer compression function with AES-256 as the underlying cipher. Here I is the 128-bit IV, and M is a 256-bit message block. A pseudo-collision satisfies the following equality:

$$H^E(I, M) = H^E(I \oplus \Delta_I, M \oplus \Delta_M)$$

Let us rewrite it:

$$E_M(I) \oplus I = E_{M\oplus\Delta_M}(I \oplus \Delta_I) \oplus I \oplus \Delta_I \;\Leftrightarrow$$
$$\Leftrightarrow\; E_M(I) \oplus E_{M\oplus\Delta_M}(I \oplus \Delta_I) = \Delta_I. \quad (11)$$

While finding many pseudo-collisions with different Δ_I, Δ_M can be done using the birthday paradox, the same task for fixed Δ_I, Δ_M is hard. This problem can be expressed as finding a solution of

$$E_{M_1}(P_1) \oplus E_{M_1\oplus\Delta_M}(P_1 \oplus \Delta_I) = E_{M_2}(P_2) \oplus E_{M_2\oplus\Delta_M}(P_2 \oplus \Delta_I) = \cdots =$$
$$= E_{M_q}(P_q) \oplus E_{M_q\oplus\Delta_M}(P_q \oplus \Delta_I) = \Delta_I, \quad (12)$$

which is harder than finding a differential q-multicollision for $E_K(\cdot)$, because the ciphertext difference is unrestricted in (1). Therefore, Lemma 1 gives us a lower bound on the complexity of this attack.

Corollary 1. *To construct q pseudo-collisions (12) with fixed Δ_I, Δ_M for an ideal cipher with an n-bit block an adversary needs at least $O(q \cdot 2^{\frac{q-2}{q+2}n})$ queries on average.*

Theorem 2. *For AES-256 in the Davies-Meyer mode q pseudo-collisions (12) can be found in time* $q \cdot 2^{67}$.

Proof. Pseudo-collision attack on the Davies-Meyer mode requires the difference in the plaintext P to be equal to the difference in the ciphertext C. The Davies-Meyer feed-forward would then cancel this difference. Our differential trail needs only to be slightly modified for this purpose. The first round of the new trail is shown in Fig. 3 (the actual values are given in Appendix A); the other rounds are the same.

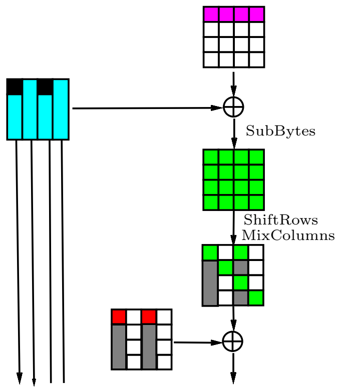

Fig. 3. The first round of the differential trail for the attack on AES-256 in the Davies-Meyer mode.

The resulting trail has 41 active S-boxes. The triangulation algorithm covers the same active S-boxes as in the proof of Theorem 1 and outputs 18 free variables. The complexity is thus the same.

Corollary 2. *AES-256 can not be used to instantiate the Davies-Meyer hashing mode.*

We expect that similar results can be shown for the other blockcipher-based constructions which are provably secure in the ideal-cipher model [3].

4 Related-key attack on AES-256

In this section we demonstrate two results: a related-key distinguisher and a key-recovery attack based on this distinguisher for the full AES-256.

4.1 Distinguisher

The first two rounds of the multicollision trail can easily be modified to build a related-key distinguisher with relatively few active S-boxes — see Appendix A for the details of the trail. The resulting trail has 19 active S-boxes in the internal states. The difference propagates through 14 S-boxes with probability $2^{-6\cdot 14}$, and through the remaining five with prob. $2^{-7\cdot 5}$. Therefore, we get a distinguisher with probability $2^{-(14\cdot 6+5\cdot 7)} = 2^{-119}$. However, the trail has five additional active S-boxes (each with probability 2^{-7}) in the key schedule. As a result, the distinguisher works for 1 out of 2^{35} related-key pairs on the average.

4.2 Key recovery

There are several ways how our trail can be used for the full 256-bit key recovery. Here we present one possibility. Steps of the attack are illustrated in Fig. 4 and are described below.

First step. We change the trail (see Fig. 4 or Appendix) to get more active S-boxes in the first two rounds which allows us to recover the key bytes at the entrance to these S-boxes. A new trail has eight active S-boxes in the first round in a "checkerboard pattern" and two in the second round. Our goal is to find ten key bytes $K^0_{0,0}$, $K^0_{0,2}$, $K^0_{1,1}$, $K^0_{1,3}$, $K^0_{2,0}$, $K^0_{2,2}$, $K^0_{3,1}$, $K^0_{3,3}$, $K^1_{0,0}$, $K^1_{0,2}$ so we execute the following procedure for each of the 2^{35} related-key pairs:

1. Repeat 2^{31} times:
   (a) Compose two structures of 2^{64} plaintexts as specified below.
   (b) Encrypt the 1st structure with K and the 2nd with $K \oplus \Delta_K$.
   (c) Sort the ciphertexts and check for a pair with the difference Δ_C.
   (d) Save a good pair if it is found.
2. For each candidate pair derive 2^{16} variants for the ten key bytes (see details below).
3. Pick the key candidate with the maximal number of votes.

The overall complexity of this procedure is $2^{31+65} = 2^{96}$ data and time, and 2^{64} memory, and it finds 80 bits of the key.

Each structure has all the possible values in bytes $\{P_{i,j}, (i+j) \mod 2 = 0\}$ (these bytes line up in columns 0 and 2 after the ShiftRows of the 1st round). The other bytes are random constants with the constraint that the two structures are related by Δ_P as in the distinguisher. For a fixed key, each structure contains 2^{64} pairs with the proper differences after the S-boxes of the 2nd round (yellow differences in two bytes). The remaining active S-boxes in rounds 3–14 require probability 2^{-93} for the trail to be fulfilled. Each structure produces a right pair of ciphertexts with probability $2^{64-93} = 2^{-29}$. Thus 2^{31} structures produce on average 4 right pairs and $2^{31+128-128} = 2^{31}$ wrong pairs.

Due to the uniform differential properties of AES S-boxes each active S-box for which we know input and output differences would suggest to us two candidates for the key byte of this S-box. For a candidate pair we guess two byte differences[3] at $B^1_{0,0}$ and $B^1_{0,2}$. We know the difference in the remaining three bytes of each column, since they should cancel out with the differences coming from the key. Thus we can undo the MixColumns for each column which allows us to know the output differences of eight S-boxes of the 1st round. We know the input differences for these S-boxes from the plaintext. In addition the two guessed difference bytes serve as inputs to the two active S-boxes of the 2nd round. The output differences for these S-boxes are known from the trail. Thus we have 10 S-boxes for which we know input and output differences which gives us $2^{10} \cdot 8 \cdot 8 = 2^{16}$ possibilities for 80-bits of the key per candidate pair. The 2^{31} wrong pairs would suggest $2^{16} \cdot 2^{31} = 2^{47}$ random keys, while the four good ones would all vote for the correct 80-bit key and some random keys. No wrong key guesses survive this step and we get 80 key bits as a result.

Second step. We proceed with changing top rounds of the trail to derive other key bytes. We remove the 2^{-6} condition on the input to the active S-box (0,0) of the 3rd round. Then we get five active S-boxes in the second round and 16 active S-boxes in the first round. We prepare 2^{90} pairs with the ciphertext difference as in the trail and decrypt them. We will try to detect pairs ($2^{90-87} = 8$ on the average) that pass the conditions in rounds 3–14. We partially encrypt all the resulting plaintext pairs using the known 80-bits and check whether the columns $\Delta B^1_{*,0}$ and $\Delta B^1_{*,2}$ follow the trail. This is a 48-bit filter on the pairs and thus we are left with 2^{42} candidate pairs. Then we guess the differences in bytes $B^1_{1,1}$ and $B^1_{3,3}$ (eight possibilities due to impossible input/output constraints in five corresponding S-boxes of the 1st and 2nd rounds), and undo the MixColumns.[4]. As a result, we have $2^{42+6} = 2^{48}$ key candidates from all pairs counting on a 64+16 = 80 bit key (64 from K^0 and 16 from bytes $K^1_{1,1}, K^1_{3,3}$). No wrong key guesses survive this step. We find the remaining 12 bytes of K^1 by exhaustive search in 2^{96} steps. The total complexity of the attack is thus dominated by the $2^{96+35} = 2^{131}$ complexity of the first step.

[3] For each byte difference there are only 8 possibilities that would not contradict with the five known differences in the 1st and 2nd rounds.

[4] We use the knowledge of the key byte $K^1_{0,0}$ to find the differences in the green diagonal at the 2nd round.

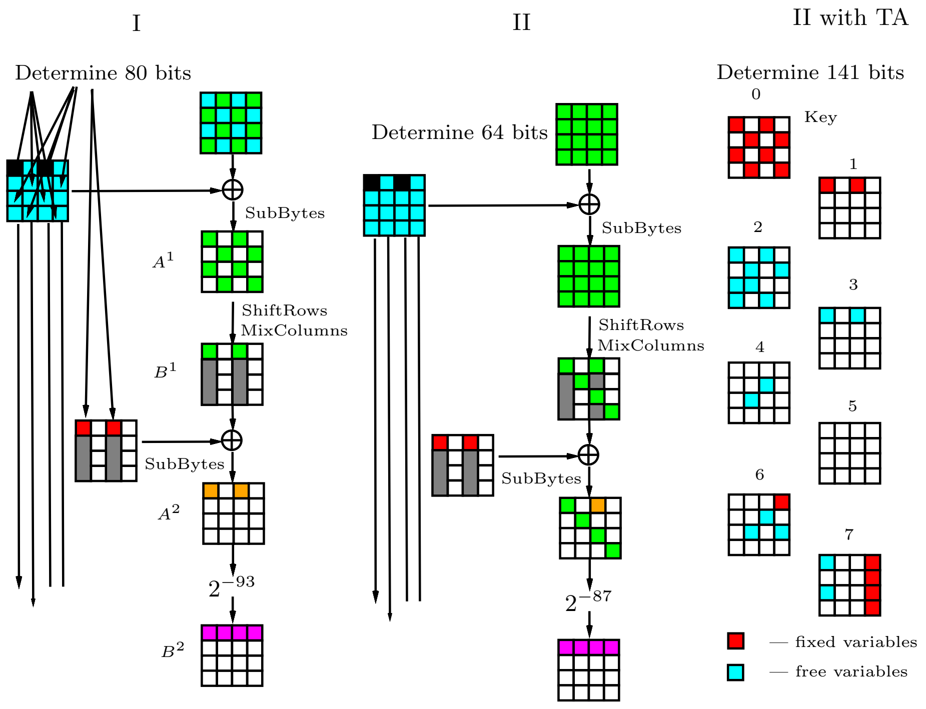

Fig. 4. Key recovery steps.

Second step with TA. If we want to avoid the chosen-ciphertext framework, there is a way to combine the knowledge of 80 key bits on top and 35 key bits in the middle with a bit higher complexity. The problem is that the fixed bits are positioned far from one another (3 key schedule rounds apart), so it seems hard to make an efficient exhaustive search on the remaining part of the key.

We solve this problem by running the triangulation algorithm on the key schedule only, where 15 bytes are marked as fixed. The algorithm outputs 17 bytes in different subkeys as free variables. Then we assign these variables randomly, choose admissible values for the remaining ones, and thus define the key guess. There are $2^{17\cdot 8+5} = 2^{141}$ possible assignments, which would determine the complexity of the key recovery.

5 Discussion and future directions

It may be intuitively clear that a cipher is not as secure as expected if some pattern (or property, or distinguisher) for it can be found much faster than for an ideal cipher even if the attacker has full access to the key (i.e. knows the key or can even choose the key). However, finding a proper definition that captures this intuition has been a challenging task for the last ten years and still is an open problem. In [2,5] it has been shown that one can construct primitives secure in random oracle model (ROM) or ideal-cipher model (ICM) but insecure when instantiated with any real-life function. However such counterexamples use the description of the fixed primitive, work for any such primitive and are thus in some sense trivial.

For instance, an adversary can randomly choose the key K, precompute $c = E_K(0)$, and then claim to have "found" a key for a pair $(0, c)$. Such "attacks" may be called trivial since they can be trivially carried out on any block cipher. Situation is trickier than in this simple example, since if the key is known or chosen by the attacker he can peel of a few rounds of the cipher and attack the remaining rounds with some non-trivial distinguisher. If we push it to the extreme and allow the attacker to peel of all the rounds we are back to our first example, which is obviously trivial. However if the attacker removes just a few rounds is such attack trivial?

Intuitively we need a challenge in the chosen or known key scenario in which even the constructor of the challenge could participate. This is similar to Blaze's challenge for the DES [4]: *Find a DES key such that some ciphertext block of the form $< XXXXXXXX >$ decrypts to a plaintext block of the form $< YYYYYYYY >$*. This challenge, then, has the desirable property that a result "speaks for itself" in demonstrating the weakness of DES, without the need for an "honest broker" who must safeguard the solution. The solution keys could not be known to any people who have not themselves searched the keyspace or found some other weakness."

The distinguisher is non-trivial if it demonstrates some rare relation between ciphertext and plaintext which has the property that the designer of the challenge could not "cook" (i.e. simulate) it in advance without doing the hard work of either breaking the cipher of finding a serious weakness in it.

6 New design criteria for block ciphers

Our results imply new design criteria for the key schedules of the block ciphers. First of all, *local collisions* should be prevented. Although it is usually easy to arrange a local collision in one round, there should be no good patterns for several rounds. A slow diffusion in the AES-256 key schedule helped us to concatenate many local collisions into a differential trail for the whole cipher. This should not be possible in a good cipher.

The key schedule should be also desynchronized with respect to the internal state. The active injection bytes in our AES trails are always located in the first row, which is not rotated. Therefore, the differences to be cancelled in a local collision should be located in the same column, or exactly 4 columns to the right in the subkey. This shift is preserved by the key schedule round, which should be certainly avoided.

Slow diffusion in the key schedule makes it also vulnerable to the triangulation algorithm or similar tools. Our preliminary analysis shows that the triangulation algorithm can cover up to two times the number of rounds needed for the full diffusion. If the key schedule in AES was ideal, we would be able to solve systems of equations on at most three-four rounds, while now we can attack five.

Finally if one considers known or chosen key attacks as a threat he needs to add two extra full diffusions on top of the number of rounds secure against standard statistical attacks.

7 Conclusions

In this paper we show a chosen key distinguisher for the 256-bit key AES with almost practical complexity of $q \cdot 2^{67}$ queries and negligible memory. It was verified by computing partial q multicollisions in time $q \cdot 2^{37}$ which takes several hours on a PC. We also show the first related-key attack on the full AES-256 with 2^{96} data and time complexity and 2^{65} memory which works for 1 out of every 2^{35} keys on average.

Acknowledgements. The authors wish to thank anonymous reviewers for their valuable comments. We also wish to thank Christian Rechberger for useful discussions. Dmitry Khovratovich is supported by PRP "Security & Trust" grant of the University of Luxembourg. Ivica Nikolić is supported by the Fonds National de la Recherche Luxembourg grant TR-PHD-BFR07-031.

References

1. Eli Biham, Orr Dunkelman, and Nathan Keller. Related-key boomerang and rectangle attacks. In *EUROCRYPT'05*, volume 3494 of *LNCS*, pages 507–525. Springer, 2005.

2. John Black. The ideal-cipher model, revisited: An uninstantiable blockcipher-based hash function. In *FSE'06*, volume 4047 of *LNCS*, pages 328–340. Springer, 2006.
3. John Black, Phillip Rogaway, and Thomas Shrimpton. Black-box analysis of the block-cipher-based hash-function constructions from PGV. In *CRYPTO'02*, volume 2442 of *LNCS*, pages 320–335. Springer, 2002.
4. Matt Blaze. Better DES challenge, 1998, available at `http://www.five-ten-sg.com/risks/risks-19.87.txt`.
5. Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.
6. Joan Daemen and Vincent Rijmen. *The Design of Rijndael. AES — the Advanced Encryption Standard*. Springer, 2002.
7. Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David Wagner, and Doug Whiting. Improved cryptanalysis of Rijndael. In *FSE'00*, volume 1978 of *LNCS*, pages 213–230. Springer, 2000.
8. Éliane Jaulmes, Antoine Joux, and Frédéric Valette. On the security of randomized cbc-mac beyond the birthday paradox limit: A new construction. In *FSE*, volume 2365 of *LNCS*, pages 237–251. Springer, 2002.
9. Antoine Joux. Multicollisions in iterated hash functions. Application to cascaded constructions. In *CRYPTO'04*, volume 3152 of *LNCS*, pages 306–316. Springer, 2004.
10. Dmitry Khovratiovich, Alex Biryukov, and Ivica Nikolić. Speeding up collision search for byte-oriented hash functions. In *CT-RSA'09*, volume 5473 of *LNCS*, pages 164–181. Springer, 2009.
11. Jongsung Kim, Seokhie Hong, and Bart Preneel. Related-key rectangle attacks on reduced AES-192 and AES-256. In *FSE'07*, volume 4593 of *LNCS*, pages 225–241. Springer, 2007.
12. Lars R. Knudsen and Vincent Rijmen. Known-key distinguishers for some block ciphers. In *ASIACRYPT'07*, volume 4833 of *LNCS*, pages 315–324. Springer, 2007.
13. National Institute of Standards and Technology (NIST). *FIPS-197: Advanced Encryption Standard*, November 2001, available at `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`.
14. National Security Agency (NSA). *National Policy on the Use of the Advanced Encryption Standard (AES) to Protect National Security Systems and National Security Information*, June 2003, available at `http://www.cnss.gov/Assets/pdf/cnssp_15_fs.pdf`.
15. Jacques Patarin. A proof of security in $O(2^n)$ for the xor of two random permutations. In *ICITS*, volume 5155 of *LNCS*, pages 232–248. Springer, 2008.
16. Kazuhiro Suzuki, Dongvu Tonien, Kaoru Kurosawa, and Koji Toyota. Birthday paradox for multi-collisions. *IEICE Transactions*, 91-A(1):39–45, 2008.

A Details on the triangulation algorithm.

The idea of the triangulation algorithm is to express all internal transformations as a set of equations. In a cipher the variables are either bytes of internal states or subkeys.

Evidently, the plaintext bytes and the key bytes completely determine all the variables. Following the framework of the Gaussian elimination process, such variables are called *free variables* since they can be assigned randomly and independently. While it is trivial to find a set of free variables when there are no constraints, it becomes harder if some variables are pre-fixed and are positioned far from one another. The fixed inputs to the active S-boxes in different rounds are an example of such case.

The algorithm iteratively searches for a variable which is involved in only one equation and can be expressed as a function of the other variables involved in that equation. If such a variable is found this implies it can be determined in the last step when all the other variables are known. Then the equation and the variable are removed from the consideration (in Gaussian elimination terminology, they are put on the diagonal), and the process goes on. An algorithm formally works as follows.

1. Build a system of equations based on the cipher description and the pre-fixed variables.
2. Mark all the variables and all the equations as non-processed.
3. Find the variable involved in only one non-processed equation. Mark the variable and the equation as processed.

4. If there exist non-processed equations go to Step 3.
5. Mark all non-processed variables as free.

If the algorithm fails then the assignment of free variables may lead to a contradiction. Evidently, the more variables are fixed *a priori*, the less likely the algorithm succeeds. On the other hand, the fewer variables we fix, the more active S-boxes would contribute to the search complexity. For AES-256, the number of free (byte) variables should be always equal to 48 minus the number of pre-fixed variables.

B Details on Trails

Davies-Meyer trail (the first two rounds):

i	Plaintext		Subkey	Ciphertext			
0	01 01 01 01		0f 0e 0f 0e	01 01 01 01			
	00 00 00 00		07 07 07 07	00 00 00 00			
	00 00 00 00		07 07 07 07	00 00 00 00			
	00 00 00 00		09 09 09 09	00 00 00 00			
i	After SB	After MC	Subkey	i	After SB	After MC	Subkey
1	30 0f 44 b0	65 00 2b 00	37 00 37 00	2	1b 00 07 00	0c 00 0e 00	0f 01 0e 00
	7c b5 93 08	1f 2c 1f 00	1f 00 1f 00		00 12 00 00	07 00 07 00	07 00 07 00
	78 d6 c2 57	1f 00 e2 00	1f 00 1f 00		00 00 1a 00	07 00 07 00	07 00 07 00
	e7 c3 29 03	21 00 21 33	21 00 21 00		00 00 00 16	09 00 09 00	09 00 09 00

Related-key distinguisher (the first two rounds):

i	Plaintext		Subkey	Ciphertext			
0	0e 0e 0e 0e		0f 0e 0f 0e	01 01 01 01			
	07 07 07 07		07 07 07 07	00 00 00 00			
	07 07 07 07		07 07 07 07	00 00 00 00			
	09 09 09 09		09 09 09 09	00 00 00 00			
i	After SB	After MC	Subkey	i	After SB	After MC	Subkey
1	1f 00 1f 00	3e 00 3e 00	37 00 37 00	2	07 00 07 00	0e 00 0e 00	0f 01 0e 00
	00 00 00 00	1f 00 1f 00	1f 00 1f 00		00 00 00 00	07 00 07 00	07 00 07 00
	00 00 00 00	1f 00 1f 00	1f 00 1f 00		00 00 00 00	07 00 07 00	07 00 07 00
	00 00 00 00	21 00 21 00	21 00 21 00		00 00 00 00	09 00 09 00	09 00 09 00

B.1 Constructing a trail

In this section we describe construction of the trail for the multicollision distinguisher. Except for the first two rounds it coincides with our related-key and Davies-Meyer distinguishers.

We use the slow diffusion in the backwards direction in the AES key schedule to get a differential trail with relatively few active S-boxes, see Fig. 5. We start with a *local collision* in one round: ΔK^i injects a one-byte difference to $A^i_{0,0}$, then it is expanded by *MixColumns* to a four-byte difference in column 0. Those 4 bytes are cancelled by the addition of ΔK^{i+1}. This local collision requires that 5 of 32 bytes of $\Delta K^i||\Delta K^{i+1}$ are non-zero. See also Fig. 2. We note that column 0 in ΔK^{i+1} should be of special form: it is produced by multiplying a column $(a, 0, 0, 0)^T$ by the *MixColumns* matrix. We call resulting vectors the *MC-columns*.

We construct a trail bottom-up and start with the local collision described above. We can cover six bottom rounds with three local collisions as shown in Fig. 5. By the dark blue color we denote five active S-boxes in these six rounds. Rounds 8, 10, and 12 do not have active S-boxes. The MC-columns are grey.

The next inverted *KeySchedule* step produces differences in all columns of K^7. This implies that four S-boxes in this *KeySchedule* step are active and produce differences in column 0 of K^6. Additionally, differences in bytes $K^6_{1,0}$, $K^6_{2,0}$, and $K^6_{3,0}$ are non-zero. Since positions of differences in the subkey are not suitable for a local collision, we assume that a 4-byte difference comes from round 6 in column 0. This column cancels 3 active differences in $K^6_{1,0}$, $K^6_{2,0}$, and $K^6_{3,0}$ so that only 4 S-boxes (row 0) are active in round 7. A difference in row 0 is expanded to the MC-difference in all 4 columns in the block and is cancelled by the injection of K^7. This step has four active S-boxes both in the key and in the block.

Since we want to reduce the number of active S-boxes, the 4-byte difference that comes from round 6 is preferred to be an MC-column. Since three bytes of it are to be cancelled by the subkey addition, column 0 of ΔK^6 must coincide with some MC-column in 3 bytes. Such a column we call a *spoiled MC-column*. Note that there is an active S-box in the key schedule in round 6.

Then we proceed with building a trail and perform 3 inverted *KeySchedule* steps. Those steps produce subkeys K^0–K^5. The difference in those subkeys are MC-columns and spoiled MC-columns. Now we define the difference in the internal states of round 1–6.

Since the plaintext has zero difference, and subkey K^0 has difference in all bytes, then all the S-boxes of the first round are active. In the second round we want to have as few active S-boxes as possible so we require that a 16-byte difference in the internal state collapses to a 5-byte difference after the addition of K^1: four active S-boxes are situated on the diagonal and one is in $A^1_{0,2}$.

After the second *MixColumns* transformation the difference in the internal state consists of one spoiled and one usual MC-column: columns 0 and 2. Rounds 3, 4, 5, and 6 have the same difference in the internal state (in the truncated form) as rounds 9, 11, 13 and 13, respectively.

At the end, we add the final AES round. It does not contain the *MixColumns* transformation so it is not possible to arrange a local collision here. The subkey K^{14} has equal differences in all the bytes of row 0. Since round 14 has no active S-boxes, those differences will appear in the ciphertext. The resulting trail has 41 active S-boxes, of these 21 are in the two first rounds.

The actual differences. The actual difference values should satisfy conditions imposed by the S-box properties, the presence of MC-columns in the subkeys, etc. We also require that differences in rounds 5 and 7–13 propagate through S-boxes with maximal probability — 2^{-6}. We programmed the search for the actual values and validated that those conditions do not lead to contradictions. The search has a negligible cost. We thus have defined the differences Δ_K (determined by subkeys K^0 and K^1) and Δ_C (determined by the subkey K^{14}). In the following subsections we list the actual differences in the trails.

Note on the colors. Differential trails are given in our figures in a truncated form. However, we marked distinct difference values with different colors. Thus the reader does not need to care about the actual values in order to understand how the trail is constructed.

The white cell stands for zero difference in a byte, the non-white cells stand for the non-zero differences. The same colors mean the same values except for the green, which denotes arbitrary differences. The exact relation between the colors and the values can be derived from the list of the actual differences. Grey and blue columns stand for MC-columns. In a spoiled MC-column one byte is marked with another color.

Differential trail for finding multicollisions:

i	Plaintext		Subkey	Ciphertext			
0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		0f 0e 0f 0e 07 07 07 07 07 07 07 07 09 09 09 09	01 01 01 01 00 00 00 00 00 00 00 00 00 00 00 00			
i	After SB	After MC	Subkey	i	After SB	After MC	Subkey
1	30 5c e1 b0 7c b5 ed 72 a6 d6 c2 16 82 eb 29 03	65 00 02 00 1f 25 1f 00 1f 00 e2 00 21 00 21 33	37 00 37 00 1f 00 1f 00 1f 00 1f 00 21 00 21 00	2	1b 00 07 00 00 12 00 00 00 00 1a 00 00 00 00 16	0c 00 0e 00 07 00 07 00 07 00 07 00 09 00 09 00	0f 01 0e 00 07 00 07 00 07 00 07 00 09 00 09 00
3	1f 1f 00 00 00 00 00 00 00 00 00 00 00 00 00 00	3e 3e 00 00 1f 1f 00 00 1f 1f 00 00 21 21 00 00	37 37 00 00 1f 1f 00 00 1f 1f 00 00 21 21 00 00	4	07 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0e 0e 00 00 07 07 00 00 07 07 00 00 09 09 00 00	0f 0e 00 00 07 07 00 00 07 07 00 00 09 09 00 00
5	1f 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	3e 00 00 00 1f 00 00 00 1f 00 00 00 21 00 00 00	37 00 00 00 1f 00 00 00 1f 00 00 00 21 00 00 00	6	07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	0e 00 00 00 07 00 00 00 07 00 00 00 09 00 00 00	0f 01 01 01 07 00 00 00 07 00 00 00 09 00 00 00
7	1f 1f 1f 1f 00 00 00 00 00 00 00 00 00 00 00 00	3e 3e 3e 3e 1f 1f 1f 1f 1f 1f 1f 1f 21 21 21 21	3e 3e 3e 3e 1f 1f 1f 1f 1f 1f 1f 1f 21 21 21 21	8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	01 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00
9	1f 00 1f 00 00 00 00 00 00 00 00 00 00 00 00 00	3e 00 3e 00 1f 00 1f 00 1f 00 1f 00 21 00 21 00	3e 00 3e 00 1f 00 1f 00 1f 00 1f 00 21 00 21 00	10	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	01 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00
11	1f 1f 00 00 00 00 00 00 00 00 00 00 00 00 00 00	3e 3e 00 00 1f 1f 00 00 1f 1f 00 00 21 21 00 00	3e 3e 00 00 1f 1f 00 00 1f 1f 00 00 21 21 00 00	12	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
13	1f 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	3e 00 00 00 1f 00 00 00 1f 00 00 00 21 00 00 00	3e 00 00 00 1f 00 00 00 1f 00 00 00 21 00 00 00	14	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	01 01 01 01 00 00 00 00 00 00 00 00 00 00 00 00

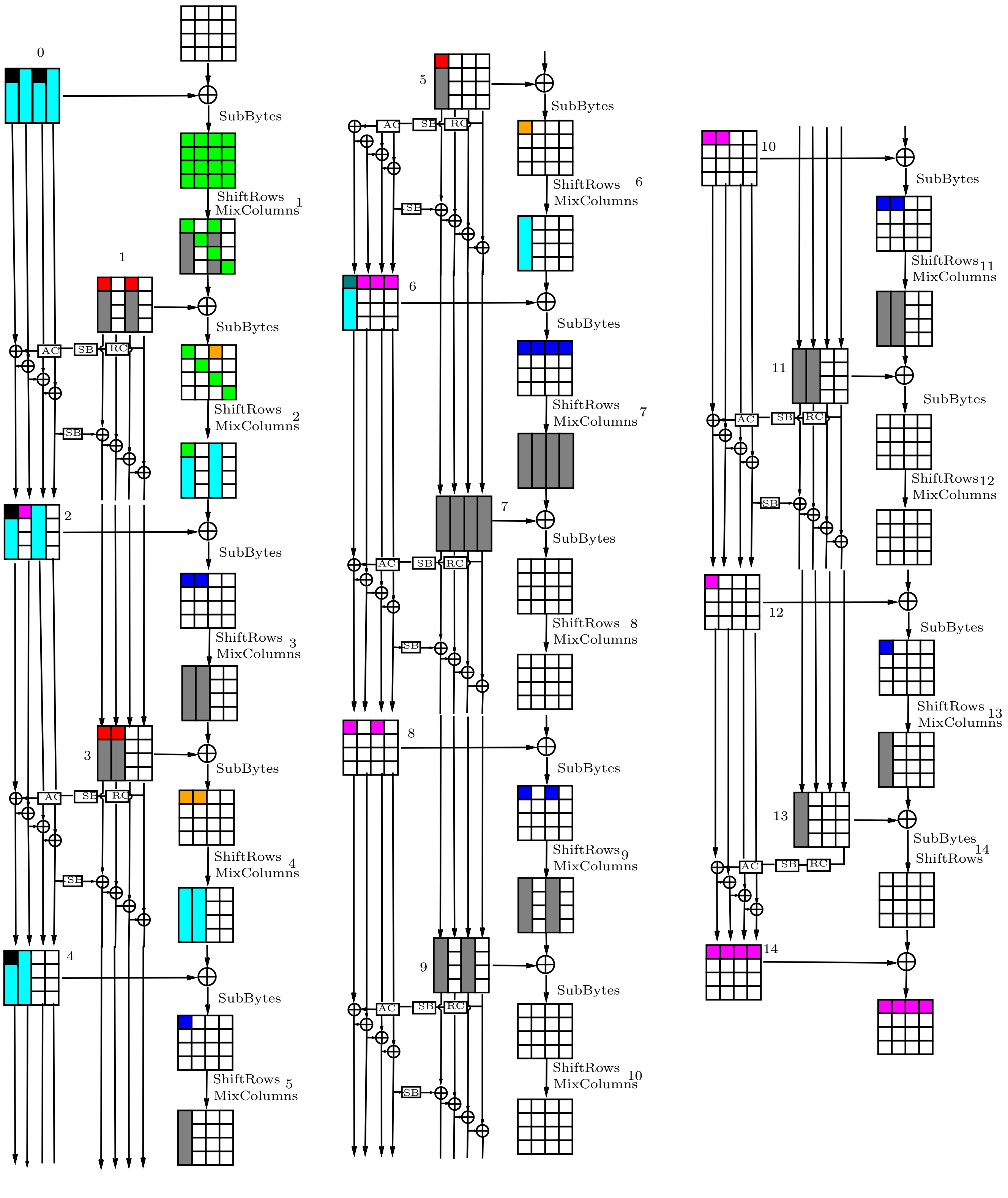

Fig. 5. Multicollision trail.