Revisiting Related-Key Boomerang attacks on AES using computer-aided tool

Patrick Derbez[1], Marie Euler[1,2], Pierre-Alain Fouque[1], Phuong Hoa Nguyen[1]

[1] Univ Rennes, CNRS, IRISA, Rennes, France
{patrick.derbez,pierre-alain.fouque,
phuong-hoa.nguyen,marie.euler}@irisa.fr
[2] Direction Générale de l'Armement, Rennes, France

Abstract. In recent years, several MILP models were introduced to search automatically for boomerang distinguishers and boomerang attacks on block ciphers. However, they can only be used when the key schedule is linear. Here, a new model is introduced to deal with non-linear key schedules as it is the case for AES. This model is more complex and actually it is too slow for exhaustive search. However, when some hints are added to the solver, it found the current best related-key boomerang attack on AES-192 with 2^{124} time, 2^{124} data, and $2^{79.8}$ memory complexities, which is better than the one presented by Biryukov and Khovratovich at ASIACRYPT 2009 with complexities $2^{176}/2^{123}/2^{152}$ respectively. This represents a huge improvement for the time and memory complexity, illustrating the power of MILP in cryptanalysis.

Keywords: AES · MILP · Boomerang Attacks

1 Introduction

Boomerang attacks have been first discovered by Wagner in [Wag99] to attack block ciphers such as Khufu, COCONUT98, FEAL, and CAST at the end of the nineties. This technique is a differential-like attack where one can merge two high probability differential trails to attack more rounds and circumvent security proofs against differential attacks. In particular, Wagner applied this technique on COCONUT98 which has been proven resistant against differential attacks using the decorrelation technique introduced by Vaudenay [Vau03]. Many techniques can be used to prove the resistance of a cipher against differential attacks, for instance the AES highlighted the wide trail strategy, but it seems more difficult to prevent boomerang or any of its variants. Consequently, recent researches on cryptanalysis focus on this powerful technique.

Patrick Derbez, Pierre-Alain Fouque and Phuong Hoa Nguyen were partially supported by the French Agence Nationale de la Recherche through the DeCrypt project under Contract ANR-18-CE39-0007.

Assuming two differentials $\alpha \to \beta$ and $\gamma \to \delta$ (with probability p and q) on two halves of the algorithm, and considering them as independent, one can build a boomerang distinguisher with probability p^2q^2. The basic idea consists in encrypting pairs of messages with a fixed difference α, assuming that a high proportion of them will follow the differential trail, and then apply another difference δ on both ciphertexts. After decrypting these new ciphertexts, one could expect to have a difference α between the plaintexts with a significant probability. Such distinguishers can be extended using standard methods to attacks recovering the secret key.

Since Wagner's paper, a series of variants of boomerang attacks (amplified boomerang, rectangle and sandwich) has drawn much attention on this technique. However, in 2011, Murphy *et al.* provided evidence on AES and DES that the independence argument was not always valid [Mur11]: some boomerang distinguishers have probability 0 *i.e.*, the boomerang never comes back. Moreover, Kircanski showed in [Kir15] using an SAT solver that some previous rectangle and boomerang attacks were based on incompatible characteristics.

Following such issues, the Boomerang Connectivity Table (BCT) has been introduced in [CHP+18] as a new tool to study the connection between the two trails. It can be seen as a precomputation of all the possible boomerangs at the S-box level. BCT solved the problem of incompatibility in boomerang distinguishers pointed out by Murphy. Since then, many improvements and further research into the BCT technique have enriched boomerang attacks.

Boomerang attacks have been applied successfully on AES [Bir04] and they are the most efficient attacks on AES-192 and AES-256 [BKN09,BK09]. At first glance, it is surprising that for these two versions, related-key boomerang attacks allow to cover all the rounds of the cipher and do not break a reduced-round version. However, it is well-known that for these versions, the diffusion in the key schedule is slow. These papers rely on the fact that there are very short local collisions if we can control the key schedule and the state of AES in a related-key attack. Such model is useful since AES was considered to construct a hash function at the end of the 2000s [GKM+09]. In [BKN09], Biryukov, Khovratovich and Nikolic present an attack on AES-256 using 2^{35} keys and 2^{96} for each key in time and data. The same year, in [BK09], Biryukov and Khovratovich improve the related key attack using only 4 keys on AES-192 with $2^{176}/2^{123}/2^{152}$ time/data/memory complexities and using 4 keys on AES-256 with $2^{99.5}/2^{99.5}/2^{77}$ time/data/memory complexities. In [BDK+10], Biryukov *et al.* show new attacks on 10-round AES-256 up to time complexity 2^{45}, data 2^{44} and memory 2^{33}, illustrating the fact that these attacks can be very effective in practice on round-reduced versions, which have been used most recently. The latter attack has been further extended by Biryukov and Khovratovich to 13 rounds with complexity 2^{76} in time, data and memory, but still in the related subkey model [BK10]. More recently, Dunkelman et al. in [DKRS20] have introduced a new technique, called retracing boomerang. By creating some correlations, they have been able to improve the probability of the distinguisher to

p^2q. They apply it on round-reduced AES and show that on 5-round AES we can recover the secret key with very low data complexity ($2^{16.5}$).

Some automatic tools have been developed for AES-128 in [BDF11] to look for low data complexity attacks on round-reduced AES. Then, more advanced differential attacks have been investigated in [DF13,DFJ13] leading to the best attack on 7-round AES-128 with time/data/memory complexity around 2^{100} and 8 and 9 rounds for AES-192 and AES-256 respectively. Li et al. have extended the former attack on AES-192 up to 9 rounds in [LJW14]. More recently, Mixed-integer Linear Programming (MILP) have been considered to automatically find boomerang distinguishers and attacks. For instance, Liu and Sasaki in [LS19], Song et al. in [SQH19] and Delaune et al. in [DDV20] have looked for the best distinguishers on the SKINNY blockcipher. In particular, Song's work shows that many boomerang distinguishers from [LGS17] against SKINNY and AES have a higher probability than expected. Finally, in [QDW+21], Qin et al. extend MILP capabilities to look for related-key rectangle attacks on SKINNY and ForkSkinny. This is a much harder problem and it has been done previously with ad-hoc tools in [BDF11,DF16] for Meet-in-the-middle (MITM) attacks on AES and impossible differential attacks on AES, mCRYPTON, SIMON, IDEA, KTANTAN, PRINCE and ZORRO. One of the most difficult task is to estimate the complexity of the attacks.

The current best time complexity for an attack on all rounds of AES-192 is $2^{189.7}$ for biclique attacks, and 2^{176} for related-key attacks. Table 1 shows some existing attacks against AES-192.

Table 1. Summary of existing attacks against AES-192. Note that biclique attacks against AES-192 are only accelerated exhaustive searches, with a complete loop on the key space.

Key Size	Rounds	Time	Data	Memory	Type	Reference
192 bits	8/12	2^{172}	2^{107}	2^{96}	MITM	[DFJ13]
	9/12	$2^{182.5}$	2^{117}	$2^{165.5}$		[LJW14]
	10/12	2^{183}	2^{124}	N/A	Related-key Rectangle	[KHP07]
		2^{156}	2^{156}	2^{65}	Related-key Differential	[GLMS18]
	12/12	2^{176}	2^{123}	2^{152}	Related-key Boomerang	[BK09]
		$2^{190.16}$	2^{80}	2^{8}	Biclique	[BKR11]
		$2^{190.83}$	2	2^{60}		[BCGS14]
		$2^{189.76}$	2^{48}	2^{60}		[TW15]
		2^{124}	2^{124}	$2^{79.8}$	Related-key Boomerang	Section 3

Our model and source codes will be publicly available at https://gitlab.inria.fr/pderbez/asia-2022-aes

Our Contributions

Looking for attacks instead of distinguishers is a harder problem. It is worth mentioning that for instance our attack on AES−192 is built from a distinguisher that has a lower probability than Biryukov's attack, but that is easier to propagate through the rest of the cipher. To this end, we have to tweak MILP models that look only for boomerang distinguishers and that work only on *linear key schedules*. When there are differences in a nonlinear key, they may not be predictable and the differences intervening in the trails cannot merely be described as free or controlled as in [DDV20]. This makes the MILP model more complex as it is discussed in Section 4.2. Finally, as mentioned before, one important feature is the computation of the probability (Section 4.3) and objective function to evaluate the cost of the attacks which is actually a rough approximation (Sections 4.4).

Then, we propose the best related-key boomerang attack on AES−192 known so far and we recover the one on AES−256 by Biryukov and Khovratovich, showing that our tool is working.

Organization of the paper

We will begin by giving an overview of AES and related key boomerang attacks in Section 2. Then we will describe our new attack on AES−192 in Section 3. Next, in Section 4, we will recall the MILP model introduced in [DDV20] to search for boomerang distinguishers and explain how we adapted it to find our attack. Finally, Section 5 concludes the paper.

2 AES and Boomerang Attacks

2.1 Description of AES

The Advanced Encryption Standard [DR02] is a Substitution-Permutation Network (SPN) that can be instantiated using three different key sizes: 128, 192, and 256. The 128-bit plaintext initializes the internal state viewed as a 4×4 matrix of bytes as values in the finite field $\mathbb{F}_{256}$, which is defined using the irreducible polynomial $x^8 + x^4 + x^3 + x + 1$ over $\mathbb{F}_2$. Depending on the version of the AES, N_r rounds are applied to that state: $N_r = 10$ for AES−128, $N_r = 12$ for AES−192 and $N_r = 14$ for AES−256. Each of the N_r AES rounds (Fig. 1) applies four operations to the state matrix (except in the last round where the **MixColumns** operation is missing):

- **AddRoundKey** (AK) adds a 128-bit subkey to the state.
- **SubBytes** (SB) applies the same 8-bit to 8-bit invertible S-Box S 16 times in parallel on each byte of the state.
- **ShiftRows** (SR) shifts the i-th row left by i positions.
- **MixColumns** (MC) replaces each of the four columns C of the state by $M \times C$ where M is a constant 4×4 maximum distance separable matrix over $\mathbb{F}_{256}$.

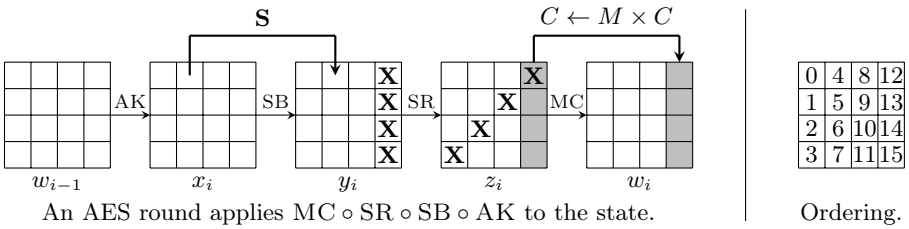

Fig. 1. Description of one AES round and the ordering of bytes in an internal state.

After the N_r-th round has been applied, a final subkey is added to the internal state to produce the ciphertext. The key expansion algorithms to produce the $N_r + 1$ subkeys are described in Fig. 2 for each keysize. We refer to the original publication [DR02] for further details.

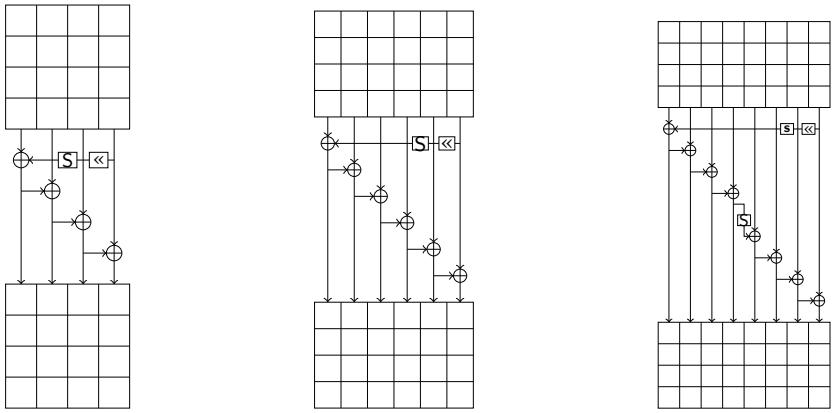

Fig. 2. Key schedules of the variants of the AES: AES-128, AES-192 and AES-256.

Notations. In this paper, we count the AES rounds from 0 and we refer to a particular byte of an internal state x by $x[i]$, as depicted in Fig. 1. Moreover, in the ith round, we denote the internal state after **AddRoundKey** by x_i, after **SubBytes** by y_i, after **ShiftRows** by z_i and after **MixColumns** by w_i. To refer to the difference in a state x, we use the notation Δx. The first added subkey is the master key k_{-1}, and the one added after round i is denoted k_i.

2.2 Probability of Boomerang Distinguishers

In a boomerang distinguisher, a cipher E is regarded as the composition of two sub-ciphers E_0 and E_1 so that $E = E_1 \circ E_0$. Suppose there exist both a

differential $\gamma \to \theta$ for E_0 and a differential $\lambda \to \delta$ for E_1 with probabilities p and q respectively.

If we assume the two differentials are independent then we obtain a boomerang distinguisher of probability:

$$\mathbb{P}\left(E^{-1}(E(P) \oplus \delta) \oplus E^{-1}(E(P \oplus \gamma) \oplus \delta) = \gamma\right) = p^2 q^2.$$

But in practice the independence assumption does not always hold, especially at the junction of both the lower and upper differentials, and many counter-examples have already been found [WKD07,Mur11,Kir15]. However, since the work of Delaune *et al.* [DDV20], we know how to precisely compute the probability of a boomerang characteristic for any SPN, assuming round independence. The idea is to compute the probability of transitions for each S-box independently, using five different tables covering all the possible cases.

Definition 1. *Given a n-bit S-box S and four differences $\gamma, \theta, \lambda, \delta \in \mathbb{F}_2^n$, the DDT, BCT [CHP+18], U(pper)/L(ower)BCT [WP19] and EBCT [DDV20] are defined as*

$$\begin{aligned}
DDT(\gamma, \theta) &= \#\{x \in \mathbb{F}_2^n \mid S(x) \oplus S(x \oplus \gamma) = \theta\} \\
BCT(\gamma, \delta) &= \#\{x \in \mathbb{F}_2^n \mid S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma\} \\
UBCT(\gamma, \theta, \delta) &= \#\left\{ x \in \mathbb{F}_2^n \;\middle|\; \begin{array}{l} S(x) \oplus S(x \oplus \gamma) = \theta \\ S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma \end{array} \right\} \\
LBCT(\gamma, \lambda, \delta) &= \#\left\{ x \in \mathbb{F}_2^n \;\middle|\; \begin{array}{l} S(x) \oplus S(x \oplus \lambda) = \delta \\ S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma \end{array} \right\} \\
EBCT(\gamma, \theta, \lambda, \delta) &= \#\left\{ x \in \mathbb{F}_2^n \;\middle|\; \begin{array}{l} S(x) \oplus S(x \oplus \gamma) = \theta \\ S(x) \oplus S(x \oplus \lambda) = \delta \\ S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma \end{array} \right\}
\end{aligned}$$

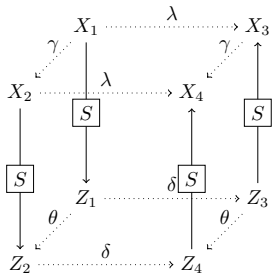

Fig. 3. Boomerang through one S-box

Fig. 3 helps to understand the notations used in these definitions. Note that all those tables are particular cases of the Extended BCT (EBCT) in which some differences are free. Intuitively, the UBCT (resp. LBCT) corresponds to the junction between the upper (resp. lower) trail and the boomerang switch, while the EBCT deals with the middle rounds of the switch. From them, one can determine the associated probabilities by dividing by 2^n. Then the probability of a boomerang distinguisher is merely the product of the probabilities of the transitions through all the S-boxes of the characteristic. For a specific S-box, the table to be used is determined by the set of its inputs/outputs which are set to a fixed value in the upper and the

lower trail. This technique does not need middle rounds to be defined, so it handles smoothly the switch between the lower and the upper differentials of a boomerang.

2.3 Boomerang Attacks on AES

In [BK09], Biryukov and Khovratovich described the first attacks against the full versions of both AES-192 and AES-256 working for all keys. These attacks are related-key boomerang attacks relying on the low diffusion in the AES key schedules.

Related keys. In the related-key model, the attacker is allowed to ask for the encryption and/or the decryption of messages under different unknown keys, related by chosen properties. More precisely, the attacker should be able to provide an algorithm $\mathcal{A}$ which takes as input a master key k_A and outputs the related keys.

For boomerang attacks, the algorithm $\mathcal{A}$ typically specifies the differences between some key bits. Because the key schedules of AES each involves non-linear S-boxes, we need to take care of the values of the differences. In particular, specifying non-zero differences both at the input and output of an S-box makes the algorithm $\mathcal{A}$ unable to generate outputs for all keys, leading to a weak-key attack. This is because there is no non-trivial differential transition through the AES S-box that holds with probability one. Regarding related-key boomerang attacks on AES, this means that the differences in the key bytes going through S-boxes in the middle of the distinguisher will most likely be null. This can be observed in both attacks as depicted in Fig. 4 and Fig. 5: the last column of almost each subkeys (as well as the fourth column for AES-256) is fully inactive for at least one of the trails.

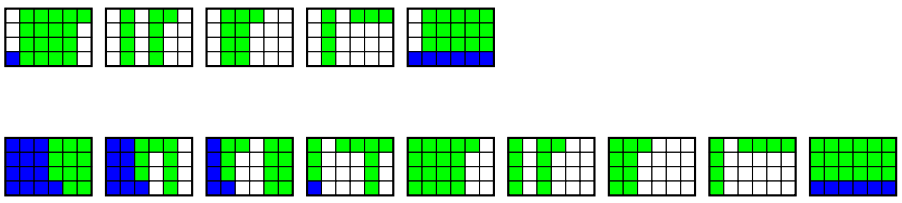

Fig. 4. Difference in the subkeys for the attack against AES-192 in [BK09]. First line is Δk, second line is ∇k. No difference are depicted in white bytes, known differences in green bytes and unknown differences in blue bytes.

Boomerang attack against AES-256. It is based on a boomerang distinguisher of probability 2^{-96} covering all rounds but the first one. The distinguisher is extended by one round at the beginning as depicted in Fig. 6. The

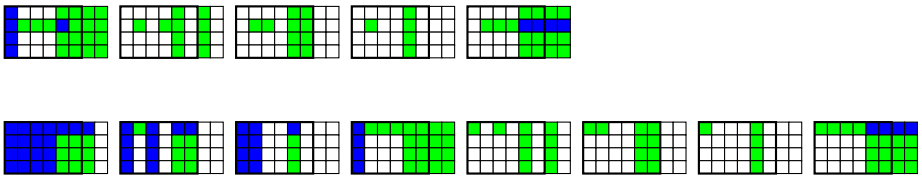

Fig. 5. Difference in the subkeys for the attack against `AES-256` in [BK09]. First line is Δk, second line is ∇k. No difference are depicted in white bytes, known differences in green bytes and unknown differences in blue bytes.

four keys k_A, k_B, k_C and k_D are generated such that they have the differences as specified in Fig. 5.

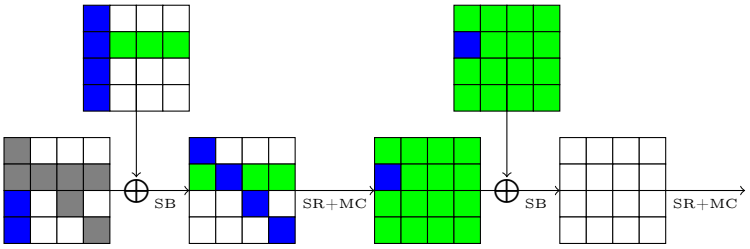

Fig. 6. Cancellations of difference in the first round of the attack against `AES-256` in [BK09]. No difference in white bytes, known differences in green bytes and unknown differences in blue and gray bytes. Differences in blue bytes fully depend on the keys and known differences.

Then, the attack procedure is quite straightforward:

1. Ask for the encryption through both k_A and k_B of a structure of $2^{9\times 8}$ plaintexts such that bytes 0, 1, 2, 3, 5, 9, 10, 13 and 15 take all the possible values while the remaining ones are constant.
2. For each plaintext p_A, look at the corresponding ciphertext c_A, apply the right difference to compute c_C and ask for its decryption under key k_C. Store the resulting plaintext p_C into a hash table indexed by the 7 constant bytes as well as $p_A[2] \oplus p_C[2]$ and $p_A[3] \oplus p_C[3]$. Indeed, while unknown, the difference in bytes 2 and 3 of the plaintexts should be equal for both pairs to satisfy the distinguisher.
3. Repeat the previous step from plaintext p_B and key k_D.
4. Look for collisions in the hash table to obtain $2^{72+72-56-16} = 2^{72}$ possible quartets.

5. For each quartet, regarding the 2 pairs (p_A, p_B) and (p_C, p_D), we know the difference at the input and at the output of $9 \times 2 = 18$ S-boxes. Each of them has on average one solution for the corresponding values and thus each quartet leads on average to one value for $18 \times 2 = 36$ bytes of key (9 for each key). Each time a value is reached, we increase a counter.
6. Because the probability of the distinguisher is 2^{-96} and the probability that one pair of the structure passes the first round is 2^{-72}, we need on average $2^{96+72-2\times72} = 2^{24}$ structures to get one right quartet. Thus we repeat the previous steps until a counter reaches the value 3 which should hardly correspond to a wrong value. This requires around $2^{25.5}$ structures.
7. The remaining key bytes are gradually recovered as detailed in [BK09].

This attack thus requires to encrypt and decrypt $2^{25.5+72} = 2^{97.5}$ messages for each of the four related keys and to process the same number of quartets leading to an overall complexity of $4 \times 2^{97.5} = 2^{99.5}$. Note that in the original attack the authors propose to perform more filtering on the quartets before increasing the counter, reducing the memory complexity to $2^{77.5}$.

Boomerang attack against AES-192. The attack against this AES version is similar to the attack against AES-256. But the key schedule of AES-192 has a better diffusion and, in particular the differences in the ciphertexts are no longer fully known. As a result, Biryukov and Khovratovich describe a procedure to recover the keys with a data complexity of 2^{123}, a time complexity of 2^{176} and a memory complexity of 2^{152}. We refer to [BK09] for further details.

3 New Attack on AES-192

In this section we describe our new attack against full AES-192 which is actually very similar to the ones of Biryukov and Khovratovich. We found a slightly better boomerang distinguisher that can be much easily turned into a key-recovery attack.

3.1 Related Keys

Generating keys k_B, k_C and k_D from an original key k_A actually relies on a boomerang distinguisher with probability 1 on the key schedule algorithm. First we note that AES key schedules are such that all subkeys can be constructed from one of them. Thus, starting from k_A we apply the chosen difference on the second subkey and compute the corresponding key k_B. Then for both k_A and k_B we apply the chosen difference on the eighth subkey and compute k_C and k_D respectively. Because the differences form a boomerang of probability 1, we can ensure that the difference between keys k_C and k_D on the second subkey is equal to the difference between keys k_A and k_B.

In our new attack, we use related keys with differences as depicted in Fig. 7. Actual values of the differences are given in Table 2.

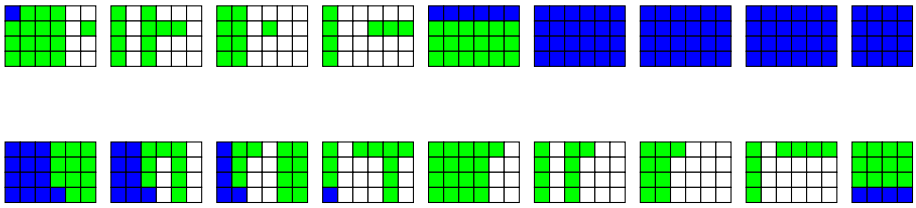

Fig. 7. Key schedule for this attack. The subkeys for the upper trail are represented above the ones of the lower trail

3.2 The Attack

Our attack is based on the boomerang trail depicted in Fig. 8. The boomerang distinguisher covers all rounds but the first and the last one and has probability $2^{-2(4\times6)} \times 2^{-2(5\times6)} = 2^{-108}$. Internal state differences are given in Appendix A. It is worth mentioning here that the boomerang distinguisher used in the original attack had probability 2^{-110}, highlighting that a small change in the distinguisher might lead to a much better attack.

The attack procedure is as follows:

1. We first observe in Fig. 7 that the differences in the last row of the ciphertext all come from the unknown difference at the output of the S-box of $k_{11}[12]$ (the active byte on the last column of the round-key before the last round-key). Thus differences in $8+3 = 11$ linear combinations of bytes are fixed. Then, because the number of bytes for which the difference is known is bigger for the plaintexts than for the ciphertexts, it is better to start the attack from the ciphertexts. Thus we first ask for the decryption under k_A of a structure of $2^{5\times8} = 2^{40}$ ciphertexts such that bytes 1, 2, 5, 6, 9, 10, 13 and 14 are constant as well as $c[3] \oplus c[7]$, $c[3] \oplus c[11]$ and $c[3] \oplus c[15]$, and while bytes 0, 3, 4, 8 and 12 take all the possible values.
2. We ask for the decryption of a similar structure under k_C, taking care that the constant values match the required difference in the 11 linear combination of bytes given above.
3. In the trail, 2 plaintext bytes have an unknown difference. Luckily, for both of them we know the expected difference after application of the S-box and thus there are only 2^{14} possible differences for the plaintexts. Hence, for each ciphertext c_A and each of the 2^{14} possible differences, we look at the corresponding plaintext p_A, apply the difference to compute p_B and ask for its encryption under key k_B. We finally store the resulting ciphertext c_B into a hash table.
4. We repeat the previous step from ciphertext c_C and key k_D.
5. We now look for collisions in the hash table on the 11 linear combinations of bytes and should obtain on average $2^{2(40+14)-11\times8} = 2^{20}$ possible quartets.

6. For each quartet, regarding the 4 pairs (p_A, p_B), (p_C, p_D), (c_A, c_C) and (c_B, c_D), we know the difference at the input and at the output of $7 \times 2 = 14$ S-boxes (2 from (p_A, p_B), 2 from (p_C, p_D), 5 from (c_A, c_C) and 5 from (c_B, c_D)). In particular, this applies a $14 - 4 = 10$-bit filter (4 Sboxes are already used at Step 3) on the quartets and thus only $2^{20-10} = 2^{10}$ of them pass this test. Each of them leads on average to 2^{14} values for 28 bytes of key (7 for each key). Each time a value is reached, we increase a counter.
7. Because the probability of the distinguisher is 2^{-108} and the probability that one pair of the structure passes the last round is 2^{-40}, we need on average $2^{108+40-2\times40} = 2^{68}$ structures to get one right quartet. Thus we repeat the previous steps until a counter reaches the value 2. This requires around 2^{69} structures.
8. The remaining key bytes are gradually recovered using a right quartet and available data.

Data complexity. In this attack we decrypt $2^{69+40} = 2^{109}$ messages under the keys k_A and k_C respectively but we encrypt $2^{69+40+14} = 2^{123}$ messages under the keys k_B and k_D.

The counter. As described in the procedure, we have to update a counter around $2^{10+14+69} = 2^{93}$ times. This can be reduced to 2^{79} by storing sequences of 14 ordered pairs of 2 bytes instead of sequences of 28 bytes. Indeed, given the input and output differences of an S-box, the symmetric of any solution for the actual values is a solution as well. Furthermore, this barely affects the success of the procedure since more than half of the sequences of 28 bytes are actually sequences of 14 ordered pairs of 2 bytes.

Noisy quartets. We generate through the procedure 2^{79} sequences of 14 ordered pairs of 2 bytes. Theoretically, there are more than $2^{8\times28-1} = 2^{223}$ such sequences. Hence, we expect on average 2^{-49} noisy quartets increasing the same counter.

Memory complexity. The hash table used during Step 5 contains $2 \times 2^{40+14} = 2^{55}$ messages. We also need to store 2^{79} sequences of 28 bytes for the counter. Thus the memory complexity is $2^{79.8}$ 128-bit blocks.

Time complexity. Filling the hash table 2^{69} times requires to process $2 \times 2^{109} + 2 \times 2^{123}$ messages and the counter is updated 2^{79} times. Regarding the missing key bytes, we used the tool developed by Bouillaguet *et al.* [BDF11] which found a procedure to enumerate all their possible values using only the constraints on the round keys in 2^{104} operations. The idea is once we know the value of the 4 keys on one byte, we know the differences on this byte in both trails. In particular we obtain the differences in some of the blue bytes leading to the knowledge of new actual values and so on. Then the 2^{104} solutions can be tested against available data.

All in all, the data complexity is $2 \times 2^{109} + 2 \times 2^{123} \approx 2^{124}$, the memory complexity is $2^{79.8}$ and the time complexity is $2 \times 2^{109} + 2 \times 2^{123} + 2^{79} \approx 2^{124}$.

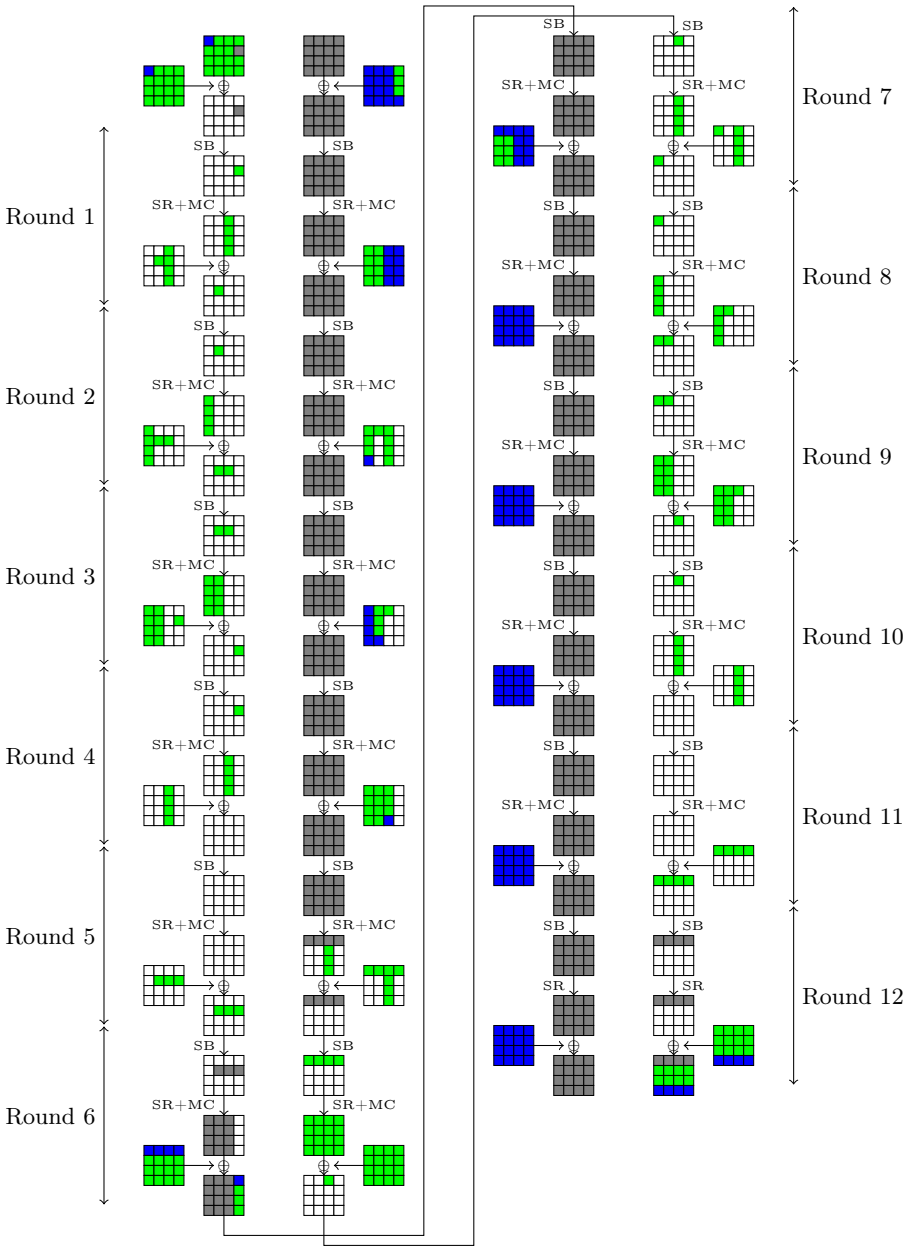

Fig. 8. Boomerang attack against `AES-192`. We recall that white stands for no difference, blue for a set difference, green for a known difference and gray for a free variable.

4 New MILP Model

Our new attack against AES-192 was found using a new MILP model dedicated to AES. In this section we thus describe this new model and discuss its limitation.

4.1 Previous Works

In 2020, both Delaune *et al.* [DDV20] and Hadipour *et al.* [HBS21] independently proposed new MILP models to search for boomerang distinguishers and applied them to the block ciphers SKINNY. Recently, at EUROCRYPT'22, Dong *et al.* [DQSW21] improved those models by adding some new constraints and a new objective function to directly search for the best rectangle attacks. Their attacks were applied to two more rounds than the previous attack on SKINNY and found better attacks on other block ciphers such as ForkSkinny, Deoxys, GIFT, Serpent.

All those models highly rely on the linearity of the key schedule and the simplicity of the internal linear layer of each target. Thus they are not well-suited to study AES. When the key schedule is nonlinear, the differences in it may be unpredictable. Therefore, the differences intervening in the trails cannot merely be described as free or controlled as in [DDV20]: some differences take a known specific value, some take an unknown (coming from the key) specific value and some are free (they can take any value uniformly). Following the previous model of [DDV20], we thus introduce a new model to search for boomerang attacks.

4.2 New Variables and Constraints

For each step and for each trail, each of the 16 differences of the state or the round key has to be described by three values answering the three following questions: is it null? is it known? is it set to a specific value? Thus for any byte a of an internal state or a round key we define three binary variables a^z, a^k, a^s containing the Boolean answers to those questions. Because we directly want to search for attacks and not only distinguishers we also add an extra binary variable a^d indicating whether the byte a belongs to the distinguisher or to the key-recovery phase.

There are several straightforward constraints involving those variables. The first and most important one is $a^z \leq a^k \leq a^s$ which states that if the difference is zero then it is known and if it is known it is set to a specific value. Furthermore, if a is a key variable then its difference cannot be free and thus $a^s = 1$. We also impose that each variable a belongs to the distinguisher in either the upper trail or the lower one which is translated into the constraint:

$$a^{d,lo} + a^{d,up} \geq 1.$$

Let us now describe precisely the constraints for each inner component of AES.

SubBytes Let $b = S(a)$ where S is the AES S-box. The first simple constraint is $a^z = b^z$ since if the input or output difference is null then both the input and output differences are null.
ShiftRows This operation being a permutation of the bytes it does not affect the variables.
MixColumns Let $(b_1, b_2, b_3, b_4) = MC(a_1, a_2, a_3, a_4)$. Because the matrix used in this operation is Maximum Distance Separable (MDS) we can simplify the constraints between the variables into

$$a_1^u + \ldots + b_4^u \in \{0, 1, 2, 3, 8\}, \text{ for } u \in \{z, k, s, d\}.$$

This can be easily translated into MILP constraints by adding an extra binary variable e and enforcing:

$$8 - a_1^u - \ldots - b_4^u \geq 5e$$
$$8 - a_1^u - \ldots - b_4^u \leq 8e$$

AddRoundKey In this operation, variables are related by an equation of the form $a \oplus b \oplus c = 0$. This corresponds to the 3 inequalities:

$$a^u - b^u - c^u \geq -1$$
$$b^u - c^u - a^u \geq -1$$
$$c^u - a^u - b^u \geq -1$$

which ensure $a^u + b^u + c^u \neq 2$ for $u \in \{z, k, s, d\}$.

We now need some constraints about the variables that belong to the distinguisher and the other ones. First we force that all key variables belong to the distinguisher. For the state variables, belonging to the distinguisher is a property propagated with probability 1. According to the notation introduced in Section 2, this means that for any r and i we have the following constraints:

$$\text{propagation through } \textbf{SubBytes}: \begin{cases} x_r[i]^{d,up} \leq y_r[i]^{d,up} \\ x_r[i]^{d,lo} \geq y_r[i]^{d,lo} \end{cases}$$
$$\text{propagation through } \textbf{MixColumns}: \begin{cases} 4z_r[i]^{d,up} \leq \sum_{j=0}^{3} w_r[4\lfloor i/4 \rfloor + j]^{d,up} \\ 4w_r[i]^{d,lo} \leq \sum_{j=0}^{3} z_r[4\lfloor i/4 \rfloor + j]^{d,lo} \end{cases}$$

In order to simplify the computation of the probability of the inner distinguisher, and more generally to simplify the whole attack, we add several extra constraints, mainly to ensure that transitions through the linear layers happen with probability 1. Here, we use the property that the constraint $a + b + c \geq 1$ only removes the solution $a = b = c = 0$ and its variants (e.g. $a + b + 1 - c \geq 1$ only removes $a = b = 1 - c = 0$). The new constraints are:

– Do not control difference outside the distinguisher:

$$\begin{cases} x_r[i]^{d,up} + 1 - x_r[i]^{s,up} + y_r[i]^{z,up} \geq 1 \\ y_r[i]^{d,lo} + 1 - y_r[i]^{s,lo} + x_r[i]^{z,lo} \geq 1 \\ w_r[i]^{d,up} + w_r[i]^{s,up} + 1 - z_r[4\lfloor i/4 \rfloor + j]^{s,up} \geq 1 & \text{for } j \in \{0,1,2,3\} \\ z_r[i]^{d,lo} + z_r[i]^{s,lo} + 1 - w_r[4\lfloor i/4 \rfloor + j]^{s,lo} \geq 1 & \text{for } j \in \{0,1,2,3\} \end{cases}$$

– Transitions through the linear layers happen with probability 1:

$$\begin{cases} 1 - z_r[i]^{d,up} + z_r[i]^{s,up} + 1 - w_r[4\lfloor i/4 \rfloor + j]^{s,up} \geq 1 & \text{for } j \in \{0,1,2,3\} \\ 1 - w_r[i]^{d,lo} + w_r[i]^{s,lo} + 1 - z_r[4\lfloor i/4 \rfloor + j]^{s,lo} \geq 1 & \text{for } j \in \{0,1,2,3\} \end{cases}$$

– Do not take back control inside the distinguisher:

$$\begin{cases} 1 - x_r[i]^{d,up} + 1 - y_r[i]^{s,up} + x_r[i]^{s,up} \geq 1 \\ 1 - y_r[i]^{d,lo} + 1 - x_r[i]^{s,lo} + y_r[i]^{s,lo} \geq 1 \\ x_r[i]^{s,up} + x_r[i]^{s,lo} \geq x_r[i]^{d,up} + y_r[i]^{d,lo} - 1 \\ y_r[i]^{s,up} + y_r[i]^{s,lo} \geq x_r[i]^{d,up} + y_r[i]^{d,lo} - 1 \end{cases}$$

Finally, as explained in Section 2.3, we want to ensure that all transitions through the keyschedule happen with probability 1. In particular, if a is a key variable and $b = S(a)$, we need to ensure that if the difference in both a and b are known then it is zero:

$$\begin{cases} 2a^{k,up} + b^{k,up} + b^{k,lo} \leq 2a^{z,up} + 2a^{z,lo} + 2 \\ 2a^{k,lo} + b^{k,up} + b^{k,lo} \leq 2a^{z,up} + 2a^{z,lo} + 2 \end{cases}$$

Note that the above inequalities involve both trails because there is no non-trivial transition through the BCT occurring with probability one.

4.3 Computing Probabilities

The probability of the inner distinguisher is computed as the product of the probability of each individual S-box transition. However, since some differences can be set but unknown, we have to extend the definitions of the BCT, UBCT, LBCT, EBCT and DDT tables. More precisely, given $b = S(a)$, we need to compute the probability of the transition for each value of $a^{z,up}$, $a^{k,up}$, $a^{s,up}$, $b^{z,up}$, $b^{k,up}$, $b^{s,up}$, $a^{z,lo}$, $a^{k,lo}$, $a^{s,lo}$, $b^{z,lo}$, $b^{k,lo}$ and $b^{s,lo}$. In practice, only 59 configurations are possible and for each of them we have to compute the associated probability. The novelty here is that some of the differences cannot be chosen to maximize the probability. For instance let consider the transition $\Delta_{in} \longrightarrow \Delta_{out}$ through the AES S-box. It is well known that we can choose $(\Delta_{in}, \Delta_{out})$ so that the probability of this transition is 2^{-6}. But now let assume that Δ_{in} is set to an unknown non-zero value and we have to choose Δ_{out}. Whatever the choice we make for it, in 126 cases the transition holds with probability 2^{-7}, in 1 case

it holds with probability 2^{-6} and in 128 cases in holds with probability 0. Translated to the distinguisher, we would be able to compute the probability that the probability of the distinguisher is not zero and, in that case, its average probability. Unfortunately, performing such precise computation for all configurations was out of reach. Instead, we only computed the average probability and would say that the transition $\Delta_{in} \longrightarrow \Delta_{out}$ holds with probability 2^{-8} in the studied case.

Overall we found 11 different possible probabilities: 2^0, $2^{-5.4}$, 2^{-6}, 2^{-8}, 2^{-12}, $2^{-13.4}$, 2^{-14}, 2^{-16}, 2^{-20}, $2^{-21.4}$ and 2^{-24}. Using classical techniques to lower the number of inequalities (mainly using the Quine-McCluskey algorithm), we were able to include the computation of the probability into our MILP model by using 5 extra binary variables and 33 inequalities per S-box.

Because the distinguisher should allow to actually distinguish the block cipher from a random permutation, we added a constraint to ensure that its probability is higher than 2^{-127}.

4.4 Objective function

Precisely evaluating the complexity of a boomerang attack is highly non-trivial and thus we chose to explore another direction. In our opinion, what matters the most for the complexity of the whole attack is on the one hand the probability of the distinguisher (p_{dist}, the $-\log_2$ of the probability) and on the other hand the number of bytes in which the differences are known in both the plaintexts and the ciphertexts. Furthermore, variables set to a specific values are more interesting than free ones since they may depend on the same unknown differences. This is actually the case in our new attack against `AES-192`. Thus, we set as objective the following expression:

$$2 \times \left(\sum_{i=0}^{15} p[i]^{k,up} + c[i]^{k,lo} \right) + 6 \times \left(\sum_{i=0}^{15} p[i]^{s,up} + c[i]^{s,lo} \right) - p_{dist},$$

and we asked the MILP solver Gurobi to maximize it. Note that we can choose other coefficients than $(2, 6)$ as long as they both are positive and sum to 8. It mainly depends on how confident we are that unknown but set differences will be related to each other.

4.5 Callback

The problem with our model is that we cannot exhaust all the possible relations between the variables. For instance, whenever 5 variables of the same column of z_r and x_{r+1} are known, a linear combination of the round key bytes is known as well. We used a callback to overcome this issue. When the MILP solver found a solution, the callback checks whether it is a valid solution, and otherwise removes

`https://www.gurobi.com/`

it via lazy constraints. We refer interested readers to [DL22] for more information regarding *lossy modelization*.

Assuming we have an equation of the form $\alpha_1 a_1 \oplus \ldots \oplus \alpha_n a_n = \beta$ where the α_i's and β are constant, we need to add to the MILP model the constraints:

$$\begin{aligned} a_1^z + \ldots + a_n^z &\neq n-1 \\ a_1^k + \ldots + a_n^k &\neq n-1 \\ a_1^s + \ldots + a_n^s &\neq n-1 \\ a_1^d + \ldots + a_n^d &\neq n-1 \end{aligned}$$

Because of the main constraints of the model, it is quite unlikely that the two last constraints are violated. However, it happens regularly for the two first ones. Checking if one such constraint is violated is actually pretty simple. We first perform a Gauss-Jordan elimination on the system of equations describing AES, echelonizing on the variables a for which $a^u = 0$ in the solution. Then we go through those equations and for each of them we check whether it satisfies the $a_1^u + \ldots + a_n^u \neq n-1$. If one equation does not, and say for instance that $a_1^u = 0$, we add the constraint $a_2^u + \ldots + a_n^u \leq n - 2 + a_1^u$ to the model.

During the callback, we check as well whether generating the keys can be done using a boomerang of probability 1. Given the system of equations, we first echelonize on state variables a for which $a^{k,up} = a^{k,lo} = 0$. Then we recursively echelonize on the key variables a for which $a^{k,up} = a^{k,lo} = 0$ and appearing *linearly* in the remaining equations (i.e. only a or $S(a)$ appears). At the end of the process, all the remaining variables should be known. Otherwise a lazy constraint is added to the model.

4.6 Limitations

Actually our model was too slow to exhaust boomerang attacks on AES. We identified two main problems:

1. When solving the relaxed problem in which all variables are not restricted to integers, Gurobi does set $a^z = a^k = a^s$ whenever it is possible. For an integer solution this would be either $(0,0,0)$ which corresponds to a free variable or $(1,1,1)$ which corresponds to a zero difference. Unfortunately, this scales badly with our constraints related to the probability of the distinguisher as this leads to a probability equals to 1. Thus the bound in Gurobi is moving very slowly.
2. Looking at the solutions for which the callback has to add a lazy constraint, we noticed that in most cases Gurobi sets a column of x_{r+1} and 3 bytes of the same column of z_r with a null difference while the corresponding column on k_r was fully set to non-zero difference (known or unknown). The problem is that on itself this configuration is possible but, in practice, it rarely passes the callback constraint regarding the keys generation process.

Thus we had to add some additional constraints to the model. More precisely, we ran the model by setting the number of active S-boxes in most of the relevant states (3 on the upper trail and 3 on the lower one). As a result, we obtained the attack against AES−192 described in Section 3. We also recovered the attack of Biryukov *et al.* against AES−256.

Note that the model is very sensitive to those extra constraints. In practice, when setting the right number of active S-boxes for 6 well-chosen states, Gurobi takes less than an hour to output the optimal pattern. But for instance if we only set the number of active S-boxes to be at most 3 (for the same 6 states), then Gurobi was still far from the optimal pattern after few days. Thus we believe it is worth improving the modelization of the problem to ensure the boomerang attack we found against AES−192 is truly optimal.

5 Conclusion

In this paper we described a new related-(sub)keys attack against full AES−192. Its complexity is 2^{52} times lower than the original attack of Biryukov and Khovratovich published at ASIACRYPT'09 while relying on a slightly better distinguisher. This highlights once again that directly searching the attack is very important as distinguishers with similar probabilities might lead to key-recovery attacks with very different complexities. Contrary to AES−256, AES−192 has a faster diffusion which makes the search of such attacks harder and is a good testbed for our tool.

We also described the MILP model which helped us to find this attack. We believe this model can still be improved a lot and opens an interesting research direction regarding automatic search of boomerang attacks with nonlinear key schedule.

References

BCGS14. Andrey Bogdanov, Donghoon Chang, Mohona Ghosh, and Somitra Kumar Sanadhya. Bicliques with minimal data and time complexity for AES. In Jooyoung Lee and Jongsung Kim, editors, *Information Security and Cryptology - ICISC 2014 - 17th International Conference, Seoul, Korea, December 3-5, 2014, Revised Selected Papers*, volume 8949 of *Lecture Notes in Computer Science*, pages 160–174. Springer, 2014.

BDF11. Charles Bouillaguet, Patrick Derbez, and Pierre-Alain Fouque. Automatic search of attacks on round-reduced AES and applications. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 169–187. Springer, 2011.

BDK+10. Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the*

Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer, 2010.

Bir04. Alex Biryukov. The boomerang attack on 5 and 6-round reduced AES. In Hans Dobbertin, Vincent Rijmen, and Aleksandra Sowa, editors, *Advanced Encryption Standard - AES, 4th International Conference, AES 2004, Bonn, Germany, May 10-12, 2004, Revised Selected and Invited Papers*, volume 3373 of *Lecture Notes in Computer Science*, pages 11–15. Springer, 2004.

BK09. Alex Biryukov and Dmitry Khovratovich. Related-key cryptanalysis of the full AES-192 and AES-256. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2009.

BK10. Alex Biryukov and Dmitry Khovratovich. Feasible attack on the 13-round AES-256. *IACR Cryptol. ePrint Arch.*, page 257, 2010.

BKN09. Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. Distinguisher and related-key attack on the full AES-256. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, 2009.

BKR11. Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique cryptanalysis of the full AES. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 344–371. Springer, 2011.

CHP+18. Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang connectivity table: A new cryptanalysis tool. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, volume 10821 of *Lecture Notes in Computer Science*, pages 683–714. Springer, 2018.

DDV20. Stéphanie Delaune, Patrick Derbez, and Mathieu Vavrille. Catching the fastest boomerangs application to SKINNY. *IACR Trans. Symmetric Cryptol.*, 2020(4):104–129, 2020.

DF13. Patrick Derbez and Pierre-Alain Fouque. Exhausting demirci-selçuk meet-in-the-middle attacks against reduced-round AES. In Shiho Moriai, editor, *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*, pages 541–560. Springer, 2013.

DF16. Patrick Derbez and Pierre-Alain Fouque. Automatic search of meet-in-the-middle and impossible differential attacks. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 157–184. Springer, 2016.

DFJ13. Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved key recovery attacks on reduced-round AES in the single-key setting. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 371–387. Springer, 2013.

DKRS20. Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. The retracing boomerang attack. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 280–309. Springer, 2020.

DL22. Patrick Derbez and Baptiste Lambin. Fast MILP models for division property. *IACR Trans. Symmetric Cryptol.*, 2022(2):289–321, 2022.

DQSW21. Xiaoyang Dong, Lingyue Qin, Siwei Sun, and Xiaoyun Wang. Key guessing strategies for linear key-schedule algorithms in rectangle attacks. *IACR Cryptol. ePrint Arch.*, page 856, 2021.

DR02. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

GKM+09. Praveen Gauravaram, Lars R. Knudsen, Krystian Matusiewicz, Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. Grøstl - a SHA-3 candidate. In Helena Handschuh, Stefan Lucks, Bart Preneel, and Phillip Rogaway, editors, *Symmetric Cryptography, 11.01. - 16.01.2009*, volume 09031 of *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, 2009.

GLMS18. David Gérault, Pascal Lafourcade, Marine Minier, and Christine Solnon. Revisiting AES related-key differential attacks with constraint programming. *Inf. Process. Lett.*, 139:24–29, 2018.

HBS21. Hosein Hadipour, Nasour Bagheri, and Ling Song. Improved rectangle attacks on SKINNY and CRAFT. *IACR Trans. Symmetric Cryptol.*, 2021(2):140–198, 2021.

KHP07. Jongsung Kim, Seokhie Hong, and Bart Preneel. Related-key rectangle attacks on reduced AES-192 and AES-256. In Alex Biryukov, editor, *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg, March 26-28, 2007, Revised Selected Papers*, volume 4593 of *Lecture Notes in Computer Science*, pages 225–241. Springer, 2007.

Kir15. Aleksandar Kircanski. Analysis of boomerang differential trails via a sat-based constraint solver URSA. In Tal Malkin, Vladimir Kolesnikov, Allison Bishop Lewko, and Michalis Polychronakis, editors, *Applied Cryptography and Network Security - 13th International Conference, ACNS 2015, New York, NY, USA, June 2-5, 2015, Revised Selected Papers*, volume 9092 of *Lecture Notes in Computer Science*, pages 331–349. Springer, 2015.

LGS17. Guozhen Liu, Mohona Ghosh, and Ling Song. Security analysis of SKINNY under related-tweakey settings (long paper). *IACR Trans. Symmetric Cryptol.*, 2017(3):37–72, 2017.

LJW14. Leibo Li, Keting Jia, and Xiaoyun Wang. Improved single-key attacks on 9-round AES-192/256. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption - 21st International Workshop, FSE 2014, London,*

UK, March 3-5, 2014. Revised Selected Papers, volume 8540 of *Lecture Notes in Computer Science*, pages 127–146. Springer, 2014.

LS19. Yunwen Liu and Yu Sasaki. Related-key boomerang attacks on GIFT with automated trail search including BCT effect. In Julian Jang-Jaccard and Fuchun Guo, editors, *Information Security and Privacy - 24th Australasian Conference, ACISP 2019, Christchurch, New Zealand, July 3-5, 2019, Proceedings*, volume 11547 of *Lecture Notes in Computer Science*, pages 555–572. Springer, 2019.

Mur11. Sean Murphy. The return of the cryptographic boomerang. *IEEE Trans. Inf. Theory*, 57(4):2517–2521, 2011.

QDW+21. Lingyue Qin, Xiaoyang Dong, Xiaoyun Wang, Keting Jia, and Yunwen Liu. Automated search oriented to key recovery on ciphers with linear key schedule applications to boomerangs in SKINNY and ForkSkinny. *IACR Trans. Symmetric Cryptol.*, 2021(2):249–291, 2021.

SQH19. Ling Song, Xianrui Qin, and Lei Hu. Boomerang connectivity table revisited. application to SKINNY and AES. *IACR Trans. Symmetric Cryptol.*, 2019(1):118–141, 2019.

TW15. Biaoshuai Tao and Hongjun Wu. Improving the biclique cryptanalysis of AES. In Ernest Foo and Douglas Stebila, editors, *Information Security and Privacy - 20th Australasian Conference, ACISP 2015, Brisbane, QLD, Australia, June 29 - July 1, 2015, Proceedings*, volume 9144 of *Lecture Notes in Computer Science*, pages 39–56. Springer, 2015.

Vau03. Serge Vaudenay. Decorrelation: A theory for block cipher security. *J. Cryptol.*, 16(4):249–286, 2003.

Wag99. David A. Wagner. The boomerang attack. In Lars R. Knudsen, editor, *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.

WKD07. Gaoli Wang, Nathan Keller, and Orr Dunkelman. The delicate issues of addition with respect to XOR differences. In Carlisle M. Adams, Ali Miri, and Michael J. Wiener, editors, *Selected Areas in Cryptography, 14th International Workshop, SAC 2007, Ottawa, Canada, August 16-17, 2007, Revised Selected Papers*, volume 4876 of *Lecture Notes in Computer Science*, pages 212–231. Springer, 2007.

WP19. Haoyang Wang and Thomas Peyrin. Boomerang switch in multiple rounds. application to AES variants and Deoxys. *IACR Trans. Symmetric Cryptol.*, 2019(1):142–169, 2019.

A New Distinguisher on AES-192

ΔK^0	?	21	21	21	00	00	ΔK^1		21	00	21	00	00	00	ΔK^2		21	21	00	00	00	00
	3e	3e	3e	3f	00	01		3e	00	3e	01	01	00		3e	3e	00	01	00	00
	1f	1f	1f	1f	00	00		1f	00	1f	00	00	00		1f	1f	00	00	00	00
	1f	1f	1f	1f	00	00		1f	00	1f	00	00	00		1f	1f	00	00	00	00
ΔK^3	21	00	00	00	00	00	ΔK^4	?	?	?	?	?	?	ΔK^5	?	?	?	?	?	?
	3e	00	00	01	01	01		3e	3e	3e	3f	3e	3f		?	?	?	?	?	?
	1f	00	00	00	00	00		1f	1f	1f	1f	1f	1f		?	?	?	?	?	?
	1f	00	00	00	00	00		1f	1f	1f	1f	1f	1f		?	?	?	?	?	?
ΔK^6	?	?	?	?	?	?	ΔK^7	?	?	?	?	?	?	ΔK^8	?	?	?	?	?	?
	?	?	?	?	?	?		?	?	?	?	?	?		?	?	?	?	?	?
	?	?	?	?	?	?		?	?	?	?	?	?		?	?	?	?	?	?
	?	?	?	?	?	?		?	?	?	?	?	?		?	?	?	?	?	?
∇K^0	?	?	?	f8	33	f8	∇K^1	?	?	33	cb	f8	00	∇K^2	?	f8	cb	00	f8	f8
	?	?	?	7c	7c	7c		?	?	7c	00	7c	00		?	7c	00	00	7c	7c
	?	?	?	7c	7c	7c		?	?	7c	00	7c	00		?	7c	00	00	7c	7c
	?	?	?	?	84	84		?	?	?	00	84	00		?	?	00	00	84	84
∇K^3	f8	00	cb	cb	33	cb	∇K^4	f8	f8	33	f8	cb	00	∇K^5	f8	00	33	cb	00	00
	7c	00	00	00	7c	00		7c	7c	7c	7c	00	00		7c	00	7c	00	00	00
	7c	00	00	00	7c	00		7c	7c	7c	7c	00	00		7c	00	7c	00	00	00
	?	00	00	00	84	00		84	84	84	84	00	00		84	00	84	00	00	00
∇K^6	f8	f8	cb	00	00	00	∇K^7	f8	00	cb	cb	cb	cb	∇K^8	f8	f8	33	f8	33	f8
	7c	7c	00	00	00	00		7c	00	00	00	00	00		7c	7c	7c	7c	7c	7c
	7c	7c	00	00	00	00		7c	00	00	00	00	00		7c	7c	7c	7c	7c	7c
	84	84	00	00	00	00		84	00	00	00	00	00		?	?	?	?	?	?

Table 2. Key schedule difference in the AES-192 trail

ΔP	? 00 00 00 00 00 00 ? 00 00 00 00 00 00 00 00	Δy^1	00 00 00 00 00 00 00 1f 00 00 00 00 00 00 00 00	Δy^2	00 00 00 00 00 1f 00 00 00 00 00 00 00 00 00 00	Δy^3	00 00 00 00 00 1f 1f 00 00 00 00 00 00 00 00 00
Δy^4	00 00 00 00 00 00 00 1f 00 00 00 00 00 00 00 00	Δy^5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Δy^6	00 00 00 00 00 ? ? ? 00 00 00 00 00 00 00 00	Δy^7	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ?
∇y^6	7c 7c 7c 7c 00 00 00 00 00 00 00 00 00 00 00 00	∇y^7	00 00 7c 00 00 00 00 00 00 00 00 00 00 00 00 00	∇y^8	7c 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	∇y^9	7c 7c 00 00 00 00 00 00 00 00 00 00 00 00 00 00
∇y^{10}	00 00 7c 00 00 00 00 00 00 00 00 00 00 00 00 00	∇y^{11}	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	∇y^{12}	? ? ? ? 00 00 00 00 00 00 00 00 00 00 00 00	ΔC	? ? ? ? 7c 7c 7c 7c 7c 7c 7c 7c ? ? ? ?

Table 3. Internal state difference in the AES-192 trail