





# The Retracing Boomerang Attack

Orr Dunkelman<sup>1</sup> , Nathan Keller<sup>2</sup> , Eyal Ronen<sup>3,4</sup>, and Adi Shamir<sup>5</sup>

<sup>1</sup> Computer Science Department, University of Haifa, Haifa, Israel

`orrd@cs.haifa.ac.il`

<sup>2</sup> Department of Mathematics, Bar-Ilan University, Ramat Gan, Israel

`nkeller@math.biu.ac.il`

<sup>3</sup> School of Computer Science, Tel Aviv University, Tel Aviv, Israel

`er@eyalro.net`

<sup>4</sup> COSIC, KU Leuven, Heverlee, Belgium

<sup>5</sup> Faculty of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, Israel

`adi.shamir@weizmann.ac.il`

**Abstract.** Boomerang attacks are extensions of differential attacks, that make it possible to combine two unrelated differential properties of the first and second part of a cryptosystem with probabilities  $p$  and  $q$  into a new differential-like property of the whole cryptosystem with probability  $p^2q^2$  (since each one of the properties has to be satisfied twice). In this paper we describe a new version of boomerang attacks which uses the counterintuitive idea of throwing out most of the data in order to force equalities between certain values on the ciphertext side. In certain cases, this creates a correlation between the four probabilistic events, which increases the probability of the combined property to  $p^2q$  and increases the signal to noise ratio of the resultant distinguisher. We call this variant a *retracing boomerang attack* since we make sure that the boomerang we throw follows the same path on its forward and backward directions. To demonstrate the power of the new technique, we apply it to the case of 5-round AES. This version of AES was repeatedly attacked by a large variety of techniques, but for twenty years its complexity had remained stuck at  $2^{32}$ . At Crypto'18 it was finally reduced to  $2^{24}$  (for full key recovery), and with our new technique we can further reduce the complexity of full key recovery to the surprisingly low value of  $2^{16.5}$  (i.e., only 90,000 encryption/decryption operations are required for a full key recovery on half the rounds of AES).

In addition to improving previous attacks, our new technique unveils a hidden relationship between boomerang attacks and two other cryptanalytic techniques, the yoyo game and the recently introduced mixture differentials.

## 1 Introduction

Differential attacks, which were introduced by Biham and Shamir [9] in 1990, use the evolution of differences between pairs of encryptions in order to construct

**Electronic supplementary material** The online version of this chapter ([https://doi.org/10.1007/978-3-030-45721-1\\_11](https://doi.org/10.1007/978-3-030-45721-1_11)) contains supplementary material, which is available to authorized users.

high probability distinguishers. They can concatenate two short differential properties with probabilities  $p$  and  $q$  into a longer property with probability  $pq$ , but only when the output difference of the first property is equal to the input difference of the second property. To overcome this restriction, Wagner [34] introduced in 1999 the idea of the boomerang attack, which “throws” two plaintexts through the encryption process, and then watches the two resultant ciphertexts (with some modifications) return back through the decryption process. This made it possible to concatenate two arbitrary differential properties whose probabilities are  $p$  and  $q$  into a longer property whose probability is  $p^2q^2$ , since it requires that four probabilistic events will simultaneously happen. This seems to be inferior to plain vanilla differential attacks, but in many cases we can find two short unrelated differential properties with much higher probabilities  $p$  and  $q$ , which more than compensates for their quadratic occurrence in  $p^2q^2$ . A typical example of the successful application of a boomerang attack is the best known related-key attack on the full versions of AES-192 and AES-256, presented by Biryukov and Khovratovich [11]. Consequently, boomerang attacks have become an essential part of the toolkit of any cryptanalyst, and many variants of this technique had been developed over the last 20 years.

In this paper we develop a new variant of the boomerang attack. We call it a *retracing boomerang attack*, since the boomerang we throw through the encryption not only returns to the plaintext side, but also follows closely related paths on its forward and backward journey. In certain cases, this makes it possible to increase the probability of the combined differential property to  $p^2q$ , since an event that happened once with probability  $q$  will reoccur a second time with probability 1. This idea had already been used by Biryukov and Khovratovich [11] in 2009 to get an extra free round in the middle of the encryption, but we use it in a different way which yields better attacks on several AES variants.

The main AES variant we consider in this paper is the 5-round version of AES. This variant had been repeatedly attacked in many papers by a large variety of techniques over the last 20 years, but all the published key recovery attacks had a complexity of  $2^{32}$  or higher. It was only in 2018 that this record had been broken, when [2] showed how to recover the full secret key<sup>1</sup> for this variant with a complexity of  $2^{24}$ . In this paper we use our new retracing boomerang attack to break the record again, reducing the complexity to  $2^{16.5}$ , albeit in the stronger attack model of adaptive chosen plaintext and ciphertext. This attack was fully verified experimentally.

Another AES variant we successfully attack is the 5-round version of AES in which the S-box and the linear mixing operations are secret key-dependent components of the same general structure as in AES. The best currently known key-recovery attack on this variant, presented by Tiessen et al. [32] in 2015, had data and time complexity of  $2^{40}$ . In this paper we show how to use our new techniques in order to reduce this complexity to just  $2^{26}$ . A comparison of our

---

<sup>1</sup> Besides the full key recovery attack, the authors of [2] present an attack with complexity of  $2^{21.5}$  that recovers 24 bits of the secret key. Since our attack recovers the full secret key, we compare it with the full key recovery attack of [2].

new attacks on 5-round AES and on 5-round AES with a secret S-box with previous attacks<sup>2</sup> is presented in Table 1.

Apart of allowing us to obtain better results in cryptanalysis of specific AES variants, our new technique unveils a hidden relation between the boomerang attack and the *yoyo tricks with AES*, introduced recently by Rønjom et al. [30]. While the ‘yoyo tricks’ differ significantly from classical boomerang attacks, we show that they fit naturally into the retracing boomerang framework. In a similar way, we show that *mixture differentials*, introduced recently by Grassi [22], is closely related to a retracing type of the rectangle attack [6, 26] (which is the chosen plaintext version of the boomerang attack). In the case of mixture differentials, the relation between the attacks is even more surprising, and may unveil additional interesting features of the mixture differential technique.

This paper is organized as follows. In Sect. 2 we present the previous related work and introduce our notations. We introduce the retracing boomerang attack in Sect. 3. We apply our new attack to 5-round AES and to 5-round AES with a secret S-box in Sects. 4 and 5, respectively. In Sect. 6 we present the retracing rectangle attack and show a relation between the mixture differential technique and the rectangle technique. We summarize the paper in Sect. 7.

## 2 Background and Previous Work

The retracing boomerang attack is related to a number of other variants of the boomerang attack, as well as to several other previously known techniques. In this section we briefly present the techniques that are most relevant to our results, while the other techniques are presented in the full version of the paper.

### 2.1 The Boomerang Attack

As the boomerang attack builds upon differential cryptanalysis, a short introduction to the latter is due.

**Differential cryptanalysis.** Introduced by Biham and Shamir [9] in 1990, differential cryptanalysis is a statistical attack on block ciphers that studies the development of differences between two encrypted plaintexts through the encryption process. Assume that we are given an iterative block cipher  $E : \{0, 1\}^n \times \{0, 1\}^k \rightarrow \{0, 1\}^n$  that consists of  $m$  (similar) rounds, and denote the intermediate value at the beginning of the  $i$ ’th round in the encryption processes of the plaintexts  $P$  and  $P'$  by  $X_i$  and  $X'_i$ , respectively. An  $r$ -round *differential characteristic* with probability  $p$  of a cipher is a property of the form  $\Pr[X_{i+r} \oplus X'_{i+r} = \Omega_O | X_i \oplus X'_i = \Omega_I] = p$ , denoted in short  $\Omega_I \xrightarrow{p} \Omega_O$ .

<sup>2</sup> We note that [4, 15, 21, 24, 25, 31] attacked an intermediate variant, in which only the S-box is key-dependent, while MixColumns is the same one as in AES. The best currently known attack on this variant, obtained by Bardeh and Rønjom [4], has complexity of  $2^{32}$ . Obviously, our attack applies to this variant as well.

**Table 1.** Attacks on 5-round AES (full key recovery)

Attack	Data (Chosen plaintexts)	Memory (128-bit blocks)	Time (encryptions)
5-Round AES			
Square [29]	$2^{11}$	small	$2^{45}$
Partial Sum [33]	$2^8$	small	$2^{40}$
Improved Square [20]	$2^{33}$	small	$2^{35}$
Imp. Diff. [7]	$2^{33.5}$	$2^{38}$	$2^{35}$
Mixture Diff. [22]	$2^{32}$	$2^{32}$	$2^{34}$
Yoyo [30]	$2^{11.3}$ ACC	small	$2^{31}$
Mixture Diff. [2]	$2^{24}$ †	$2^{21.5}$	$2^{24}$ †
<b>Our Attack</b> (Sect. 4)	$2^9$ ACC	$2^9$	$2^{23}$
<b>Our Attack</b> (Sect. 4)	$2^{15}$ ACC	$2^9$	$2^{16.5}$
5-Round AES with Secret S-boxes			
Integral [31]	$2^{128}$	small	$2^{128}$
Integral [25]	$2^{96}$	$2^8$	$2^{96}$
Imp. Diff. [24]	$2^{102}$	$2^8$	$2^{102}$
Imp. Diff. [21]	$2^{76.4}$	$2^8$	$2^{76.4}$
Mult.-of- $n$ . [21]	$2^{53.3}$	$2^{16}$	$2^{53.3}$
Square‡ [32]	$2^{40}$	$2^{36}$	$2^{40}$
Yoyo [4]	$2^{32}$ ACC	small	$2^{31}$
<b>Our Attack</b> ‡ (Sect. 5)	$2^{17.5}$ ACC	$2^{17}$	$2^{29}$
<b>Our Attack</b> ‡ (Sect. 5)	$2^{25.8}$ ACC	$2^{17}$	$2^{25.8}$

†—the data and time complexity for partial key recovery is  $2^{21.5}$

‡—the attack applies also when the linear transformation is key-dependent

ACC—Adaptive Chosen Plaintexts and Ciphertexts

Differential cryptanalysis shows that if there exists a differential characteristic for most of the rounds of the cipher that holds with a non-negligible probability, then the cipher can be broken faster than exhaustive search by an attack that requires  $O(1/p)$  chosen plaintexts. Differential cryptanalysis was used to mount the first attack faster than exhaustive search on the full DES [28], as well as on many other block ciphers.

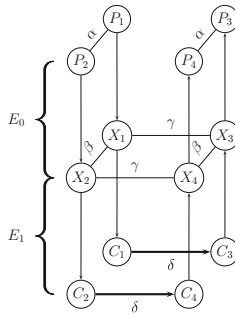
**The boomerang attack.** Introduced by Wagner [34], the boomerang attack was one of the first techniques to show that non-existence of ‘long’ high-probability differentials is not sufficient to guarantee security with respect to differential-type attacks. Suppose that the cipher  $E$  can be decomposed as  $E = E_1 \circ E_0$ , such that for  $E_0$ , there exists a differential characteristic  $\alpha \xrightarrow{p} \beta$ , and for  $E_1$ , there exists a differential characteristic  $\gamma \xrightarrow{q} \delta$ , depicted in Fig. 1, where  $pq \gg 2^{-n/2}$ . Then one can distinguish  $E$  from a random permutation, using Algorithm 1 presented below.

---

**Algorithm 1.** The Boomerang Attack Algorithm

---

- 1: Initialize a counter  $ctr \leftarrow 0$ .
  - 2: Generate  $(pq)^{-2}$  unique plaintext pairs  $(P_1, P_2)$  with input difference  $\alpha$ .
  - 3: **for all** pairs  $(P_1, P_2)$  **do**
  - 4:     Ask for the encryption of  $(P_1, P_2)$  to  $(C_1, C_2)$ .
  - 5:     Compute  $C_3 = C_1 \oplus \delta$  and  $C_4 = C_2 \oplus \delta$ . ▷  $\delta$ -shift
  - 6:     Ask for the decryption of  $(C_3, C_4)$  to  $(P_3, P_4)$ .
  - 7:     **if**  $P_3 \oplus P_4 = \alpha$  **then**
  - 8:         Increment  $ctr$
  - 9:     **if**  $ctr \geq 1$  **then**
  - 10:         **return:** This is the cipher  $E$ .
  - 11:     **else**
  - 12:         **return:** This is a random permutation.
- 



**Fig. 1.** The boomerang attack

The theoretical analysis of the algorithm is as follows. Denote the intermediate values after the partial encryption by  $E_0$  of the plaintext  $P_j$  by  $X_j$ , for  $1 \leq j \leq 4$ . Let  $(P_1, P_2)$  be a plaintext pair such that  $P_1 \oplus P_2 = \alpha$ . By the differential characteristic of  $E_0$ , we have

$$X_1 \oplus X_2 = \beta, \tag{1}$$

with probability  $p$ . On the other side, as the ciphertexts satisfy  $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ , by the differential characteristic of  $E_1$  we have

$$(X_1 \oplus X_3 = \gamma) \wedge (X_2 \oplus X_4 = \gamma), \tag{2}$$

with probability  $q^2$ . (We recall that the differential characteristic  $\gamma \xrightarrow{q} \delta$  for  $E_1$  is identical to the differential characteristic  $\delta \xrightarrow{q} \gamma$  for  $E_1^{-1}$ , in the sense that both count the same set of input/output pairs for  $E_1$ .) If both Eq. (1) and (2) hold, then we have

$$X_3 \oplus X_4 = (X_3 \oplus X_1) \oplus (X_1 \oplus X_2) \oplus (X_2 \oplus X_4) = \gamma \oplus \beta \oplus \gamma = \beta. \tag{3}$$

Therefore, by the differential characteristic of  $E_0$ , we have  $P_3 \oplus P_4 = \alpha$ , with probability  $p$ . Hence, assuming (somewhat non-carefully, as discussed in [27]) that all these events are independent, we have

$$\Pr[P_3 \oplus P_4 = \alpha | P_1 \oplus P_2 = \alpha] = p^2 q^2. \quad (4)$$

As we take  $1/(pq)^2$  pairs  $(P_1, P_2)$ , then with a high probability ( $= 1 - e^{-1} \approx 63\%$ ),<sup>3</sup> for at least one of them we obtain  $P_3 \oplus P_4 = \alpha$ , and hence, the algorithm outputs ‘the cipher  $E$ ’. On the other hand, for a random permutation we have  $\Pr[P_3 \oplus P_4 = \alpha] = 2^{-n}$ , and hence, the expected number of pairs  $(P_1, P_2)$  for which  $P_3 \oplus P_4 = \alpha$  holds is  $2^{-n} \cdot (pq)^{-2} \ll 1$  (as we assumed  $pq \gg 2^{-n/2}$ ). Thus, with an overwhelming probability, the algorithm outputs ‘random permutation’.

Therefore, the above algorithm indeed allows distinguishing  $E$  from a random permutation, using in total  $4(pq)^{-2}$  adaptively chosen plaintexts and ciphertexts (in the sequel: ACPC).

## 2.2 The S-Box Switch

In [11], Biryukov and Khovratovich showed that in certain cases, the boomerang attack can be improved significantly by ‘bypassing for free’ some operations in the middle of the cipher. One of those cases, called *S-box switch*, is particularly relevant to our results. Assume that  $E = E_1 \circ E_0$ , where the last operation in  $E_0$  is a layer  $S$  of S-boxes applied in parallel (which is the usual scenario in SP networks, like the AES). That is,  $S$  divides the state  $\rho$  into  $\rho = (\rho_1, \rho_2, \dots, \rho_t)$  and transforms it to  $S(\rho) = (f_1(\rho_1) || f_2(\rho_2) || \dots || f_t(\rho_t))$ , for  $t$  independent (keyed) functions  $f_j$ . Suppose that the differential characteristics in  $E_0, E_1$  are such that in both  $\beta$  and  $\gamma$ , the difference in the part of the intermediate state  $X$  that corresponds to the output of some  $f_j$  is  $\Delta$ . In other words, denoting this part of the intermediate state  $X$  by  $X_j$ , if both characteristics hold then we have

$$(X_1)_j \oplus (X_2)_j = (X_1)_j \oplus (X_3)_j = (X_2)_j \oplus (X_4)_j = \Delta.$$

In such a case, we have  $(X_1)_j = (X_4)_j$  and  $(X_2)_j = (X_3)_j$ , and hence, if the differential characteristic in the function  $(f_j)^{-1}$  holds for the pair  $(X_1, X_2)$  then it must hold for the pair  $(X_3, X_4)$ . Thus, the overall probability of the boomerang distinguisher is increased by a factor of  $(q')^{-1}$ , where  $q'$  is the probability of the differential characteristic in  $f_j$ .

This ‘switch’, along with other ‘switches in the middle’, was a key ingredient in the attack of [11] on the full AES-192 and AES-256. Later on, some of these switches were generalized in the Sandwich attack of [19] for the case of a probabilistic transition in the middle layer and used to attack KASUMI, the cipher of 3G cellular networks. Recently, a more complete and rigorous analysis of the transition between  $E_0$  and  $E_1$  was suggested, using the Boomerang Connectivity Table [14] that covers these and related ideas. These developments are described in more detail in the full version of the paper.

<sup>3</sup> The success probability of the attack can be increased by slightly enlarging the data complexity. If we start with  $c/(pq)^2$  pairs, then the success probability is  $1 - e^{-c}$ .

### 2.3 The Yoyo Game and Mixture Differentials

In addition to the classical boomerang attack, two more techniques – the yoyo game and mixture differentials – are closely related to our attacks. We describe them very briefly below, but in more detail in the sequel. Our new type of boomerang attacks allows us to unveil a close relation of these two techniques to the boomerang and rectangle techniques, respectively.

**The yoyo game.** The yoyo technique was introduced by Biham et al. [5] in 1998. Like the boomerang attack, the yoyo game is based on encrypting a pair of plaintexts  $(P_1, P_2)$ , modifying the corresponding ciphertexts  $(C_1, C_2)$  into  $(C_3, C_4)$ , and decrypting them. However, while the boomerang distinguisher just checks whether the resulting plaintexts  $(P_3, P_4)$  satisfy some property, in the yoyo game the process continues:  $(P_3, P_4)$  are modified into  $(P_5, P_6)$  which are encrypted into  $(C_5, C_6)$ , those in turn are modified into  $(C_7, C_8)$  which are decrypted into  $(P_7, P_8)$ , etc. The process satisfies the property that all *pairs of intermediate values*  $(X_{2\ell+1}, X_{2\ell+2})$  at some specific point of the encryption process satisfy some property (e.g., zero difference in some part of the state). Since for a random permutation, the probability that such a property is satisfied by a sequence of pairs  $(X_{2\ell+1}, X_{2\ell+2})$  is extremely low, this property can theoretically be used for distinguishing the cipher from a random permutation. Practically, exploiting this property is not so easy, as the adversary does not see the intermediate values  $(X_{2\ell+1}, X_{2\ell+2})$ . Nevertheless, Biham et al. showed that in some specific cases, such a distinguishing is possible and even allows for key recovery [5].

Biham et al. [5] applied the yoyo technique to a 16-round variant of the block cipher Skipjack. Biryukov et al. [12] applied it to attack generic 5-round Feistel constructions, and Rønjom et al. [30] used it to attack reduced-round AES with at most 5 rounds. As the attack of Rønjom et al. [30] is a central ingredient in our attacks on 5-round AES, it is presented in detail in Sect. 4.

**Mixture differentials.** The mixture differential technique was presented by Grassi [22]. The technique works in the following setting. Assume that the cipher  $E$  can be decomposed as  $E = E_1 \circ E_0$ , where  $E_0$  can be considered as a concatenation of several permutations, i.e.,  $P = (\rho_1, \rho_2, \dots, \rho_t)$  and  $E_0(P) = f_1(\rho_1) || f_2(\rho_2) || \dots || f_t(\rho_t)$ , for  $t$  independent functions  $f_j$ . A well known example of such  $E_0$  is 1.5 rounds of AES, that can be treated as four parallel super S-boxes (see [16]).

**Definition 1.** *Given a plaintext pair  $(P^1, P^2)$ , where  $P^1 = (\rho_1^1, \dots, \rho_t^1)$  and  $P^2 = (\rho_1^2, \dots, \rho_t^2)$  we say that  $(P^3, P^4)$ , where  $P^3 = (\rho_1^3, \dots, \rho_t^3)$  and  $P^4 = (\rho_1^4, \dots, \rho_t^4)$  is a mixture counterpart of  $(P^1, P^2)$  if for each  $1 \leq j \leq t$ , the quartet  $(\rho_j^1, \rho_j^2, \rho_j^3, \rho_j^4)$  consists of two pairs of equal values or of four equal values. The quartet  $(P^1, P^2, P^3, P^4)$  is called a mixture.*

The main observation behind the mixture differential technique is that if  $(P^1, P^2, P^3, P^4)$  is a mixture then the XOR of the corresponding intermediate values  $(X^1, X^2, X^3, X^4)$  is zero. Indeed, for each  $j$ , as  $(\rho_j^1, \rho_j^2, \rho_j^3, \rho_j^4)$  consists of two pairs

of equal values, then the same holds for  $(f_j(\rho_j^1), f_j(\rho_j^2), f_j(\rho_j^3), f_j(\rho_j^4))$  as well. In particular,  $f_j(\rho_j^1) \oplus f_j(\rho_j^2) \oplus f_j(\rho_j^3) \oplus f_j(\rho_j^4) = 0$ . As a result, if we have  $X^1 \oplus X^3 = \gamma$ , then  $X^2 \oplus X^4 = \gamma$  holds as well. Now, if there exists a differential characteristic  $\gamma \xrightarrow{q} \delta$  for  $E_1$ , then with probability  $q^2$ , the corresponding ciphertexts satisfy  $C^1 \oplus C^3 = C^2 \oplus C^4 = \delta$ .

Grassi [22, 23] applied the technique to mount several attacks on AES with up to 6 rounds. The 5-round attack of Grassi was recently improved in [2] into an attack with overall complexity of  $2^{24}$  for full key-recovery (or  $2^{21.5}$  for recovering 24 bits of the secret key), that is significantly faster than all other known attacks on 5-round AES.

### 3 The Retracing Boomerang Attack

Our new *retracing boomerang* framework contains two attack types – a *shifting* type and a *mixing* type. In this section we present these two types and discuss their advantages over the standard boomerang attack and their relation to previous works. In the following sections and in the appendix we present applications of the new techniques, along with a few variants and extensions.

#### 3.1 The Shifting Retracing Attack

**Assumptions.** Suppose that the block cipher  $E$  can be decomposed as  $E = E_{12} \circ E_{11} \circ E_0$ , where  $E_{12}$  consists of dividing the state into two parts (a left part of  $b$  bits and a right part of  $n - b$  bits) and applying to them the functions  $E_{12}^L, E_{12}^R$ , respectively. Furthermore, suppose that for  $E_0$ , there exists a differential characteristic  $\alpha \xrightarrow{p} \beta$ , for  $E_{11}$ , there exists a differential characteristic  $\gamma \xrightarrow{q_1} (\mu_L, \mu_R)$ , for  $E_{12}^L$ , there exists a differential characteristic  $\mu_L \xrightarrow{q_2^L} \delta_L$ , and for  $E_{12}^R$ , there exists a differential characteristic  $\mu_R \xrightarrow{q_2^R} \delta_R$  (see Fig. 2).<sup>4</sup>

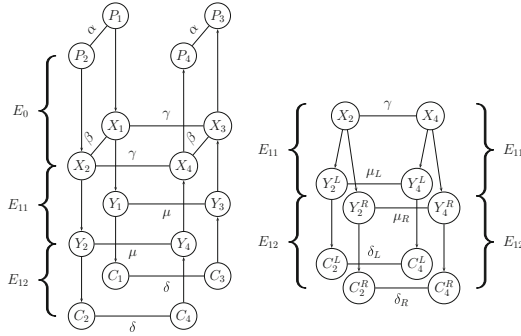
In other words, we make the same assumptions as in the boomerang attack, with the additional assumption that  $E_1$  can be further decomposed into two sub-ciphers, and that the second sub-cipher has a specific structure. While this additional assumption may look very restrictive, it applies for a wide class of block ciphers. For example, if  $E$  is a SASAS construction [13], then  $E_{12}$  can be taken to be the last  $S$  layer; a specific such example is AES [29], where  $E_{12}$  can be taken to be the last 1.5 rounds.

**The attack procedure and its analysis.** Assuming that  $p q_1 q_2^L q_2^R \gg 2^{-n/2}$ , the standard boomerang attack can be used to distinguish  $E$  from a random permutation, with data complexity of  $4(p q_1 q_2^L q_2^R)^{-2}$ .

The basic idea of the retracing boomerang attack is to add an artificial  $(b-1)$ -bit filtering in the middle of the attack procedure. Namely, after encrypting

<sup>4</sup> A variant of the attack that is applicable when the top part of the cipher can be further decomposed into two sub-ciphers, is presented in the full version of the paper.



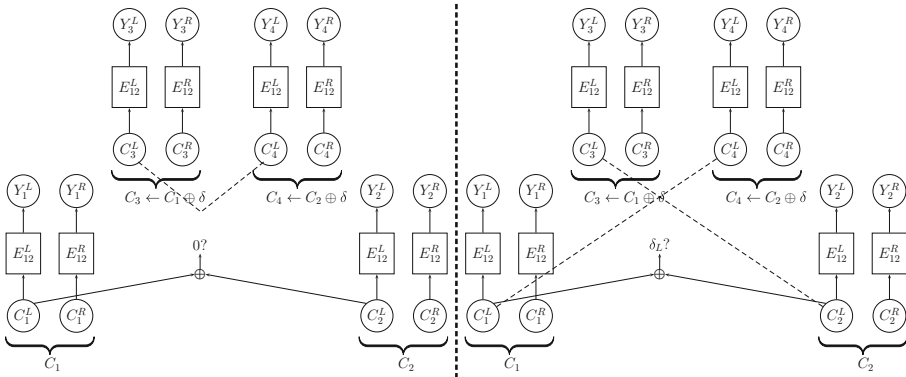


**Fig. 2.** The retracing boomerang framework

$(P_1, P_2)$  into  $(C_1, C_2)$ , we first check whether

$$C_1^L \oplus C_2^L = 0 \text{ or } \delta_L. \tag{5}$$

Only if there is equality, we continue with the boomerang process. Otherwise, we simply discard the pair  $(P_1, P_2)$ . See Fig. 3 for the process.



**Fig. 3.** A shifted quartet (dashed line means equality)

This is a surprising move, as the discarded pair may actually be a *right pair* with respect to the differential characteristic  $\alpha \rightarrow \beta$  (i.e., a pair that satisfies the characteristic). Hence, a natural question arises: *What do we gain from this filtering?*

Note that for any value of  $\delta_L$ , if Eq. (5) holds then the two unordered pairs  $(C_1^L, C_3^L)$  and  $(C_2^L, C_4^L)$  are identical. Hence, if one of these pairs satisfies the differential characteristic  $\delta_L \xrightarrow{q_2^L} \mu_L$ , then the other one *must* satisfy it as well. As a result, the probability of the boomerang distinguisher *among the examined pairs* is increased by a factor of  $(q_2^L)^{-1}$  from  $(pq_1q_2^Lq_2^R)^2$  to  $(pq_1q_2^R)^2q_2^L$ .

**Advantages of the new technique.** At first glance, our new variant of the boomerang attack seems completely odd and useless. Note that as the block size of  $E_{12}^L$  is  $b$  bits, then any possible differential characteristic of  $E_{12}^L$  has probability of at least  $2^{-b+1}$ , and so, the overall probability of the boomerang distinguisher is increased by a factor of at most  $2^{b-1}$ . On the other hand, our filtering leaves only  $2^{-b+1}$  of the pairs, so we either gain nothing (if  $q_2^L = 2^{-b+1}$ ) or even lose (otherwise)!

It turns out that there are several advantages to this approach:

1. *Improving the signal to noise ratio.* Recall that the ordinary boomerang attack applies if  $pq_1q_2^Lq_2^R \gg 2^{-n/2}$ , as otherwise, the probability that  $P_3 \oplus P_4 = \alpha$  holds for  $E$  is not larger than the respective probability for a random permutation. In the retracing boomerang attack, the probability that  $P_3 \oplus P_4 = \alpha$  holds *among the examined pairs* is increased by a factor of  $(q_2^L)^{-1}$ , while the probability for a random permutation remains unchanged. As a result, the attack can succeed in cases where the ordinary boomerang attack fails due to insufficient filtering.

Furthermore, the adversary can use the increased gap between the probabilities of the checked event for  $E$  and for a random permutation to replace the differential characteristic  $\beta \xrightarrow{p} \alpha$  used for the pair  $(X_3, X_4)$  in the backward direction with a truncated differential characteristic.  $\beta \xrightarrow{p'} \alpha'$  of a higher probability  $p'$  in which  $\alpha'$  specifies the difference in only some part of the bits, while still having a larger probability of the event  $P_3 \oplus P_4 = \alpha'$  for  $E$  than for a random permutation. An example of this advantage is demonstrated in the attack on 5-round AES presented in the full version of the paper.

2. *Reducing the data complexity.* The new attack saves data complexity on the decryption side. Indeed, as decryption is performed only to the pairs that satisfy the filtering condition, the number of decryptions is reduced by a factor of  $2^{b-1}$ . While usually, the effect of this reduction is not significant as then the encryptions dominate the overall complexity, there are cases in which more queries are made on the decryption side, and in such cases, the data complexity may be reduced significantly. This advantage (like the previous one) is demonstrated in the attack on 5-round AES in the full version of the paper.

3. *Reducing the time complexity.* The smaller number of pairs on the decryption side may affect also the time complexity of the attack. This effect is not significant when the attack complexity is dominated by encryption/decryption of the data. However, in many cases (e.g., where a round is added before the distinguisher and the adversary has to guess some key material in the added round and check whether the condition  $P_3 \oplus P_4 = \alpha$  holds), the complexity of the attack is dominated by analysis of the pairs  $(P_3, P_4)$ . In such cases, the time

complexity may be reduced by a factor of  $(q_2^L)^{-1}$ , as the number of pairs  $(P_3, P_4)$  is reduced by this ratio.

**Relation to previous works.** Our new technique uses several ideas that already appeared in previous works in different contexts. Those include:

- *Discarding part of the data before the analysis.* The counter-intuitive idea of neglecting part of the data appears in various previous works, e.g., in the context of time-memory tradeoff attacks on stream ciphers [18], and in the context of conditional linear attacks on DES [8].
- *Increasing the probability of the boomerang attack by exploiting dependency between differentials.* As we mentioned above, several previous works on the boomerang attack used dependency between differentials, and in particular, situations in which the four inputs to some function in the encryption process are composed of two pairs of equal values, to increase the probability of the boomerang distinguisher (see, e.g., [10, 11, 14, 19]). The closest to our attack is the *S-box switch* of Biryukov and Khovratovich [11] described in Sect. 2. In all these attacks, the gain is obtained *in the transition between the two sub-ciphers*  $E_0, E_1$ . In contrast, the retracing boomerang exploits dependency between the two differentials in the same sub-cipher (by forcing dependency via the artificial filtering).
- *Increasing the probability of the boomerang attack by exploiting representation of a sub-cipher as two (or more) functions applied in parallel.* Such a probability increase was obtained by Biryukov and Khovratovich [11] in the *ladder switch* technique, which exploits a subdivision into multiple functions (e.g., a layer of S-boxes) along with dependency between differentials, to increase the probability of the transition between the two sub-ciphers.
- *Using quartets of the form  $(x, x, y, y)$  to force dependency.* This idea was recently used by Grassi in [22, Theorem 4], in the context of the mixture differential attack described in Sect. 2.

### 3.2 The Mixing Retracing Attack

**The attack setting.** Recall that the shifting retracing boomerang attack increases the probability of the boomerang distinguisher by forcing equality between the unordered pairs  $(C_1^L, C_2^L)$  and  $(C_3^L, C_4^L)$  that enter  $(E_{12}^L)^{-1}$ . Such an equality can be forced in an alternative way, without inserting an artificial filtering.

Instead of working with the same shift  $\delta$  for all ciphertexts, one may shift each ciphertext pair  $(C_1, C_2)$  by  $(C_1^L \oplus C_2^L, 0)$ , thus obtaining the ciphertexts

$$C_3 = (C_3^L, C_3^R) = (C_1^L \oplus (C_1^L \oplus C_2^L), C_1^R \oplus 0) = (C_2^L, C_1^R),$$

and (similarly)  $C_4 = (C_1^L, C_2^R)$ , see Fig. 4. In such a case, the unordered pairs  $(C_1^L, C_3^L)$  and  $(C_2^L, C_4^L)$  are equal, and hence, we gain a factor of  $(q_2^L)^{-1}$ , like in the shifting retracing attack. Furthermore, in the right part we have  $C_1^R = C_3^R$

and  $C_2^R = C_4^R$ , and thus, we gain also a factor of  $(q_2^R)^{-2}$  (as both characteristics in  $E_{12}^R$  hold trivially with probability 1). This results in a total gain of  $(q_2^L)^{-1}(q_2^R)^{-2}$ .

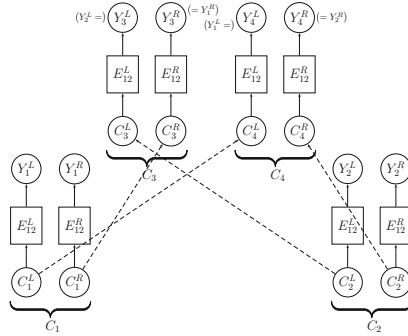


Fig. 4. A mixture quartet of ciphertexts (a dashed line means equality)

**Relation to ‘yo-yo tricks with AES’.** Interestingly, in the special case of the AES, the mixing described here is exactly the core step of the yo-yo attack of Rønjom et al. [30] (presented in detail in Sect. 4). Hence, this type of retracing boomerang is not entirely novel, but rather generalizes and presents a new viewpoint on the yo-yo attack of Rønjom et al.

**Comparison between the two types of retracing boomerang.** At first glance, it seems that the mixing retracing attack is clearly better than the shifting retracing attack presented above. Indeed, it obtains an even larger gain in the probability of the distinguisher, while not discarding ciphertext pairs! However, there are several advantages of the shifting variant that make it more beneficiary in various scenarios:

- *Using structures.* A central technique for extending the basic boomerang attack is adding a round at the top of the distinguisher, using structures. This technique can be combined with the shifting retracing technique, as follows. First, the adversary performs the ordinary boomerang attack with structures (i.e., encrypts structures of plaintexts, shifts all ciphertexts by  $\delta$  and decrypts the resulting ciphertexts), and then she checks the artificial filtering together with the condition on  $P_3, P_4$ , since both can be checked simultaneously using a hash table. As a result, the data complexity remains the same as in the ordinary boomerang attack (with structures!), while the retracing boomerang leads to an improvement in the signal to noise ratio,

which can be translated to a reduction in the data complexity, as described above.

For mixing retracing, such a combination is impossible, since each ciphertext pair  $(C_1, C_2)$  has to be modified by its own shift  $(C_1^L \oplus C_2^L, 0)$ , and so, one cannot shift entire structures as a single block. Therefore, the reduction of data complexity by using structures cannot be obtained.

A similar advantage of the shifting variant is the ability to combine it with extension of the boomerang attack by adding a round at the bottom, as we demonstrate in our attack on 6-round AES in the full version of the paper.

- *Combination with  $E_{11}$* . In the mixing variant, since the output difference for  $(E_{12}^L)^{-1}$  (namely,  $(C_1)^L \oplus (C_2)^L$ ), is arbitrary and changes between different pairs, in most cases there is no good combination between differential characteristics of  $(E_{12}^L)^{-1}$  that can be used and differential characteristics of  $(E_{11})^{-1}$ . Indeed, in the yoyo attack of [30] on 5-round AES, this part of the attack succeeds simply because  $E_{11}$  is empty. It seems that while the mixing retracing technique can be applied also in cases where  $E_{11}$  is non-linear (and, in particular, non-empty), it will usually (or even almost always) be inferior to the shifting retracing boomerang in such cases.
- *Construction of ‘friend pairs’*. An important ingredient in many boomerang attacks is ‘friend pairs’, which are pairs that are attached to given pairs in such a way that if some pair satisfies a desired property then all its ‘friend pairs’ satisfy the same property as well (such pairs are used in most attacks in this paper). While both types of the retracing boomerang attack allow constructing several ‘friend pairs’ for each pair, the number of pairs in the shifting variant is significantly larger, which makes it advantageous in some cases.

**The names of the attacks.** The shifting type of the retracing boomerang is named this way since it preserves the  $\delta$ -shift of the original boomerang attack, and uses the filtering to enhance the probability of the original boomerang process. The mixing type is named this way since it replaces the  $\delta$ -shift by a mixing procedure, like the one used in mixture differentials [22].

## 4 Retracing Boomerang Attack on 5-Round AES

Our first application of the retracing boomerang framework is an improved attack on 5-round AES, which allows recovering the full secret key with data complexity of  $2^{15}$ , time complexity of  $2^{16.5}$ , and memory complexity of  $2^9$ . The attack was fully implemented experimentally. Since our attack is based on central components of the yoyo attack of Rønjom et al. [30] on 5-round AES (which can be seen as a mixing retracing boomerang attack, as was shown in Sect. 3.2), we begin this section with describing the structure of the AES and presenting the attack of [30]. Then we present our attack, its analysis, and its experimental verification.

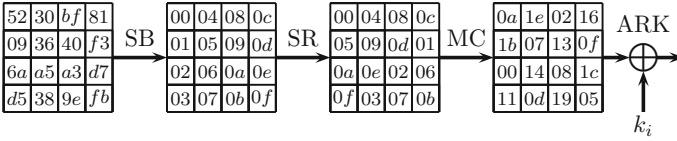


Fig. 5. An AES round

### 4.1 Brief Description of the AES and Notations

The Advanced Encryption Standard (AES) [29] is a substitution-permutation (SP) network which has 128-bit plaintexts and 128, 192, or 256-bit keys. Since the descriptions of all attacks we present in this paper are independent of the key schedule, we do not differentiate between these variants.

The 128-bit internal state of AES is treated as a byte matrix of size  $4 \times 4$ , where each byte represents a value in  $GF(2^8)$ . An AES round (described in Fig. 5) applies four operations to this state matrix:

- SubBytes (SB)—applying the same 8-bit to 8-bit invertible S-box 16 times in parallel on each byte of the state,
- ShiftRows (SR)—cyclically shifting the  $i$ 'th row by  $i$  bytes to the left,
- MixColumns (MC)—multiplication of each column by a constant  $4 \times 4$  matrix over the field  $GF(2^8)$ , and
- AddRoundKey (ARK)—XORing the state with a 128-bit subkey.

An additional AddRoundKey operation is applied before the first round, and in the last round the MixColumns operation is omitted. The number of rounds is between 10 and 14, depending on the key length. We omit the key schedule, as it does not affect the description of our attacks.

The bytes of each state of AES are numbered  $0, 1, \dots, 15$ , where for  $0 \leq i, j \leq 3$ , the  $j$ 'th byte in the  $i$ 'th row is numbered  $i + 4j$  (see the state after SB in Fig. 5). We always consider 5-round AES, where the MixColumns operation in the last round is omitted. The rounds are numbered  $0, 1, 2, 3, 4$ . The subkeys are numbered  $k_{-1}, k_0, \dots, k_4$ , where  $k_{-1}$  is the secret key XORed to the plaintext at the beginning of the encryption process. We denote by  $W, Z$ , and  $X$  the intermediate states before the MixColumns operation of round 0, at the input to round 1 and before the MixColumns operation of round 2, respectively. The  $j$ 'th byte of a state or a key  $X_i$  is denoted by  $X_{i,j}$  or by  $(X_i)_j$ . When several bytes  $j_1, \dots, j_\ell$  are considered simultaneously, they are denoted by  $X_{i,\{j_1, \dots, j_\ell\}}$  or by  $(X_i)_{\{j_1, \dots, j_\ell\}}$ .

The term ' $\ell$ 'th shifted column' (resp. ' $\ell$ 'th inverse shifted column') refers to the result of application of SR (resp.,  $SR^{-1}$ ) to the  $\ell$ 'th column. For example, the 0'th shifted column consists of bytes 0, 7, 10, 13, and the 0'th inverse shifted column consists of bytes 0, 5, 10, 15. We also denote by  $SR(j)$  (resp.,  $SR^{-1}(j)$ ) the byte position to which byte  $j$  is transformed by SR (resp.,  $SR^{-1}$ ).

When considering differences between the encryption processes of a pair of plaintexts, we say that a component (e.g., byte or column) at some stage of

the encryption process is *active* if the difference in that component is non-zero. Otherwise, we call the component *passive*. Finally, we say that some values  $x_1, x_2, \dots, x_m$  ‘sum up to zero’ if  $x_1 \oplus x_2 \oplus \dots \oplus x_m = 0$ .

## 4.2 The Yoyo Attack of Rønjom et al. on 5-Round AES

**The idea behind the attack.** The attack decomposes 5-round AES as  $E = E_{12} \circ E_{11} \circ E_0$ , where  $E_0$  consists of the first 2.5 rounds,  $E_{11}$  is the MC operation of round 2, and  $E_{12}$  consists of rounds 3 and 4. For  $E_0$  in the forward direction, the adversary uses a truncated differential characteristic whose input difference is zero in three inverse shifted columns, and whose output difference is zero in a single shifted column. The probability of the characteristic is  $4 \cdot 2^{-8} = 2^{-6}$ , since it holds if and only if the output difference of the active column in round 0 is zero in at least one byte. For  $E_{12}$  in the backward direction, recall that 1.5 rounds of AES can be represented as four 32-bit to 32-bit super S-boxes applied in parallel (see [16]). For each ciphertext pair  $(C_1, C_2)$ , the adversary modifies it into one of its mixture counterparts (see Definition 1) with respect to the division into super S-boxes, calls the new ciphertext pair  $(C_3, C_4)$ , and asks for its decryption. Due to the mixture construction, the four outputs of each super S-box are composed of two pairs of equal values, and hence, the four corresponding inputs to the super S-boxes sum up to 0. As MC is a linear operation, this implies that  $X_1 \oplus X_2 \oplus X_3 \oplus X_4 = 0$ . Therefore, with probability  $2^{-6}$ , the difference  $X_3 \oplus X_4$  equals zero in a shifted column. This, in turn, implies that the difference  $Z_3 \oplus Z_4$  equals zero in an inverse shifted column (i.e., one of the four quartets of bytes: (0, 5, 10, 15), (1, 4, 11, 14), (2, 5, 8, 15), (3, 6, 9, 12)).

At this point, the adversary would like to attack bytes 0, 5, 10, 15 of the subkey  $k_{-1}$ , using the fact that in one of the bytes of the first column, we have  $Z_3 \oplus Z_4 = 0$ . However, this information provides only an 8-bit filtering, while 32 subkey bits are involved. In order to improve the filtering, the authors of [30] construct ‘friend pairs’ of the pair  $(Z_3, Z_4)$ , such that if we have  $Z_3 \oplus Z_4 = 0$  in byte  $\ell$ , then the same holds for all friend pairs. The resulting attack algorithm (of [30]) is given in Algorithm 2.

**Analysis of the attack.** The data complexity of the attack is about  $2^9$ , since for each of  $2^6$  pairs  $(P_1, P_2)$ , the adversary decrypts four ciphertext pairs  $(C_3^j, C_4^j)$ . The time and memory complexities are dominated by the attack on  $k_{-1}$  in Step 7. In a naive application, this attack requires about  $2^{32}$  operations for each pair  $(P_1, P_2)$  and each value of  $\ell \in \{0, 1, 2, 3\}$ , and thus, the overall time complexity of the attack is about  $2^{32} \cdot 2^6 \cdot 4 = 2^{40}$ . The authors of [30] managed to improve the overall complexity to  $2^{31}$ , using a careful analysis of round 0, including exploitation of the specific matrix used in MC. We do not present this part of the attack, as it can be replaced by a simpler and stronger tool, as we describe below. To summarize, the data complexity of the attack is  $2^9$  adaptively chosen plaintexts and ciphertexts, the memory complexity is  $2^9$  and the time complexity is  $2^{31}$  encryptions.

**Algorithm 2.** Rønjom et al.'s Yoyo Attack on 5-Round AES

- 
- 1: Ask for the encryption of  $2^6$  pairs  $(P_1, P_2)$  of chosen plaintexts that have non-zero difference only in bytes 0,5,10,15.
  - 2: **for all** corresponding ciphertext pairs  $(C_1, C_2)$  **do**
  - 3:     Define four modified ciphertext pairs  $(C_3^j, C_4^j)$  ( $j = 1, 2, 3, 4$ ) to be mixture counterparts of the pair  $(C_1, C_2)$ .
  - 4:     Ask for the decryption of the ciphertext pairs and consider the pairs of intermediate values after round 0,  $(Z_3^j, Z_4^j)$ .
  - 5:     **for all**  $\ell \in \{0, 1, 2, 3\}$  **do**
  - 6:         Assume that all four pairs  $(Z_3^j, Z_4^j)$  and the pair  $(Z_1, Z_2)$  have zero difference in byte  $\ell$ .
  - 7:         Use the assumption to extract bytes 0, 5, 10, 15 of  $k_{-1}$ .
  - 8:         **if** a contradiction is reached **then**
  - 9:             Increment  $\ell$
  - 10:         **if**  $\ell > 3$  **then**
  - 11:             Discard the pair
  - 12:         **else**
  - 13:             Use the fact that  $Z_3^j \oplus Z_4^j = 0$  in the entire  $\ell$ 'th inverse shifted column to attack the three remaining columns of round 0 (sequentially) and thus to deduce the rest of  $k_{-1}$ .
- 

**4.3 A Simple Improvement of the Yoyo Attack on 5-Round AES**

A simple improvement of the attack of Rønjom et al. is to use a meet-in-the-middle (MITM) procedure to attack bytes 0, 5, 10, 15 of  $k_{-1}$  in Step 7.

Denote the intermediate value in byte  $m$  before the MC operation of round 0 in the encryption of a plaintext  $P$  by  $W_m$ . W.l.o.g. we consider the case  $\ell = 0$ , and recall that by the structure of AES, byte 0 in the input to round 1 satisfies

$$Z_0 = 02_x \cdot W_0 \oplus 03_x \cdot W_1 \oplus 01_x \cdot W_2 \oplus 01_x \cdot W_3. \quad (6)$$

In the MITM procedure, the adversary guesses bytes 0, 5 of  $k_{-1}$ , computes the value

$$02_x \cdot (W_3^j)_0 \oplus 03_x \cdot (W_3^j)_1 \oplus 02_x \cdot (W_4^j)_0 \oplus 03_x \cdot (W_4^j)_1 \quad (7)$$

for  $j = 1, 2, 3$ , and stores the concatenation of these values (i.e., a 24-bit value) in a sorted table. Then she guesses bytes 10, 15 of  $k_{-1}$ , computes the value

$$01_x \cdot (W_3^j)_2 \oplus 01_x \cdot (W_3^j)_3 \oplus 01_x \cdot (W_4^j)_2 \oplus 01_x \cdot (W_4^j)_3 \quad (8)$$

for  $j = 1, 2, 3$ , and checks for a match in the table (which is, of course, equivalent to the condition  $(Z_3^j)_0 = (Z_4^j)_0$  for  $j = 1, 2, 3$ ). As this condition is a 24-bit filtering, about  $2^{32} \cdot 2^{-24} = 2^8$  suggestions for bytes 0, 5, 10, 15 of  $k_{-1}$  remain, and those can be checked using the conditions  $(Z_3^4)_0 = (Z_4^4)_0$  and  $(Z_1)_0 = (Z_2)_0$ .

The data complexity of the attack remains  $2^9$ . The time complexity is reduced to  $2^6 \cdot 4 \cdot 2^{16} = 2^{24}$  operations, where each operation is roughly equivalent to a computation of one AES round in a single column for 6 plaintexts, or a total of less than  $2^{23}$  encryptions.



It seems that the use of MITM increases the memory complexity of the attack to about  $2^{16}$ . However, one can maintain the memory at  $2^9$  using the *dissection* technique [17] (see, e.g., [2] for similar applications of dissection). Therefore, the time complexity of the attack is reduced to  $2^{23}$  encryptions, while the data and memory complexities remain unchanged at  $2^9$ .

#### 4.4 An Attack on 5-Round AES with Overall Complexity of $2^{16.5}$

We now show how one can reduce the time complexity of the attack described above to  $2^{16.5}$ , at the expense of increasing the data complexity to about  $2^{15}$ .

The idea behind the attack is to enhance the MITM procedure, such that on each of the two sides, the number of possible key values is reduced to  $2^8$  (instead of  $2^{16}$ ). The reduction is obtained using two methods:

*Constructing an extra equation by a specific choice of plaintexts.* In order to reduce the number of possible values of  $k_{-1,\{0,5\}}$ , we choose plaintext pairs with non-zero difference only in bytes 0, 5. For such pairs, the condition  $(Z_1)_0 = (Z_2)_0$  simplifies into

$$02_x \cdot (W_1)_0 \oplus 03_x \cdot (W_1)_1 \oplus 02_x \cdot (W_2)_0 \oplus 03_x \cdot (W_2)_1, \quad (9)$$

as bytes 2, 3 of  $W$  cancel out. This equation depends only on the plaintexts and on bytes 0, 5 of  $k_{-1}$ , and since it is an 8-bit filtering, it leaves only  $2^8$  possible values of  $k_{-1,\{0,5\}}$ . In order to detect these  $2^8$  candidates efficiently, we make our choice of plaintexts even more specific.

We choose only pairs of plaintexts  $(P_1, P_2)$  that satisfy  $(P_1)_5 \oplus (P_2)_5 = 01_x$ . In addition, as a precomputation phase we compute the row of the Difference Distribution Table (DDT) of the AES S-box that corresponds to input difference  $01_x$  and store it in memory, where each output difference is stored along with the value(s) that lead to it.<sup>5</sup>

As a result, for each pair  $(P_1, P_2)$  and for each guess of  $k_{-1,0}$ , we can use Eq. (9) to compute the output difference of the SB operation in byte 5. As the input difference is fixed to be  $01_x$ , we can use the precomputed row of the DDT to find the inputs to that SB operation by a single table lookup, and hence, to retrieve instantly the possible value(s) of  $k_{-1,5}$  that correspond to the guessed value of  $k_{-1,0}$ .

This process allows us to compute the  $2^8$  possible values of  $k_{-1,\{0,5\}}$  in about  $2^8$  simple operations for each pair.

*Eliminating a key byte from the equation by using multiple ‘friend pairs’.* In order to reduce the number of possible values of  $k_{-1,\{0,5\}}$ , we attach to each plaintext pair  $(P_1, P_2)$  multiple ‘friend pairs’, such that if  $(P_1, P_2)$  satisfies the differential characteristic of  $E_0$ , then all friend pairs satisfy the same characteristic as well.

<sup>5</sup> Constructing this row takes  $2^9$  simple operations, and storing it takes much less than  $2^9$  128-bit cells of memory.

We perform the boomerang process for all friend pairs together with the original pairs, obtaining many pairs  $(P_3^j, P_4^j)$ . We choose one such pair for which we have

$$(P_3^j)_{10} \oplus (P_4^j)_{10} = 0 \quad \text{or} \quad (P_3^j)_{15} \oplus (P_4^j)_{15} = 0. \quad (10)$$

Assume w.l.o.g. that the equality holds in byte 10. We perform the MITM procedure presented above with *the single pair*  $(P_3^j, P_4^j)$ . Note that the first step provided us with  $2^8$  possible values for  $k_{-1, \{0,5\}}$ . Hence, in the MITM attack there are only  $2^8$  possible values for the expression (7). On the other hand, by the choice of the pair, there is zero difference in byte 2 before the MC operation, and thus, the subkey byte  $k_{-1,10}$  cancels out from the expression (8). As a result, this expression depends on a single key byte, and thus, has only  $2^8$  possible values, just like Eq. (7). Thus, the MITM procedure requires about  $2^9$  simple operations and (as the data provides an 8-bit filtering) leaves  $2^8$  suggestions for subkey bytes  $k_{-1, \{0,5,15\}}$ . Finally, we can take any other couple of ‘friend pairs’ and recover the unique value of  $k_{-1, \{0,5,10,15\}}$  by another MITM procedure in which one side computes the contribution of bytes 0, 1, 3 to Eq. (9) (applied for the difference  $(Z_3)_0 \oplus (Z_4)_0$ ) and the other side computes the contribution of byte 2, as on each side there are about  $2^8$  possible values.

Therefore, the complexity of the MITM attack on  $k_{-1, \{0,5,10,15\}}$  is reduced to about  $2^8$  operations for each pair  $(P_1, P_2)$  and each value of  $\ell$ , or a total of about  $2^{16}$  operations. As for the data complexity, in order to have a friend pair that satisfies Eq. (10) with a high probability, we have to take about  $2^7$  friend pairs for each of the  $2^6$  pairs  $(P_1, P_2)$ . Hence, the total data complexity is increased to about  $2^{15}$ . A more precise analysis is given below.

**Attack algorithm.** The algorithm of our improved attack on 5-round AES is as follows.

1. **Precomputation:** Compute the row of the DDT of the AES S-box that corresponds to input difference  $01_x$ , along with the actual values.
2. **Online phase:** Take 64 pairs  $(P_1, P_2)$  of plaintexts such that in each pair, we have  $(P_1)_5 = 00_x$  and  $(P_2)_5 = 01_x$ , in byte 0 the values  $(P_1, P_2)$  are distinct, and in all other bytes, the values  $(P_1, P_2)$  are equal.
3. To each plaintext pair  $(P_1, P_2)$ , attach  $2^7$  ‘friend pairs’  $(P_1^j, P_2^j)$ , such that for each  $j$  we have  $(P_1^j \oplus P_2^j) = P_1 \oplus P_2$ , and  $(P_1^j)_{\{0,5,10,15\}} = (P_1)_{\{0,5,10,15\}}$ .
4. Do the following for each plaintext pair  $(P_1, P_2)$ , and for each  $\ell \in \{0, 1, 2, 3\}$ : [we present the operations for  $\ell = 0$ , the other cases are similar.]
  - (a) For each guess of byte  $k_{-1,0}$ , partially encrypt  $(P_1, P_2)$  through the SB operation in byte 0 of round 0 to find its output difference. Then, assuming that the pair  $(P_1, P_2)$  satisfies the characteristic of  $E_0$  with  $\ell = 0$  (i.e., that  $(Z_1)_0 = (Z_2)_0$ ), use Eq. (9) to find the output difference of the SB operation in byte 5 of round 0. Then use the precomputed DDT to deduce the actual inputs to that SB operation, and deduce from them the value of subkey byte  $k_{-1,5}$ . Store in a table the  $2^8$  possible values  $k_{-1, \{0,5\}}$ .
  - (b) Ask for the encryption of  $(P_1, P_2)$  and of its  $2^7$  ‘friend pairs’  $(P_1^j, P_2^j)$ . For each ciphertext pair  $(C_1, C_2)$  or  $(C_1^j, C_2^j)$  we obtain, replace it by

one of its mixture counterparts, which we denote by  $(C_3, C_4)$  or  $(C_3^j, C_4^j)$  (respectively), and ask for its decryption. Denote the resulting plaintext pairs by  $(P_3, P_4)$  and  $(P_3^j, P_4^j)$ .

- (c) Find a value  $j$  for which the pair  $(P_3^j, P_4^j)$  satisfies Eq. (10). [In the following steps we assume w.l.o.g. that the condition yields equality in byte 10. If the equality is in byte 15, the steps should be modified accordingly.]
  - (d) Perform a MITM attack on Column 0 of round 0, using the plaintext pair  $(P_3^j, P_4^j)$ . Specifically, use the  $2^8$  possible values for  $k_{-1, \{0,5\}}$  computed in Step 4(a) to obtain  $2^8$  possible values for (7) and store them in a table. Then, for each guess of subkey byte  $k_{-1,15}$ , compute (8) and check in the table for a collision. Each collision provides us with a possible value of  $k_{-1, \{0,5,15\}}$ .
  - (e) Perform a MITM attack on Column 0 of round 0, using two other plaintext pairs  $(P_3^{j'}, P_4^{j'})$ . Specifically, use the  $2^8$  possible values for  $k_{-1, \{0,5,15\}}$  computed in the previous step to obtain the contribution of bytes 0, 1, 3 to Eq. (6) (applied for the difference  $(Z_3)_0 \oplus (Z_4)_0$ , for both pairs) and store it in a table. Then, for each guess of subkey byte  $k_{-1,10}$ , compute the contribution of byte 2 to Eq. (6) and check in the table for a collision. (Each collision provides us with a possible value of  $k_{-1, \{0,5,10,15\}}$ , along with a filtering for wrong pairs.) If a contradiction is reached, move to the next value of  $\ell$ ; if contradiction is reached for all values of  $\ell$ , discard the pair  $(P_1, P_2)$  and move to the next pair.
5. Using a pair  $(P_1, P_2)$  for which no contradiction occurred in Step 4 and its ‘friend pairs’, perform MITM attacks on Columns 1, 2, and 3 of round 0 (sequentially), exploiting the fact that  $Z_3 \oplus Z_4$  equals zero in the  $\ell$ ’th inverse shifted column (e.g., for  $\ell = 0$  this column consists of bytes 0, 5, 10, 15), to recover the rest of the subkey  $k_{-1}$ .

**Attack analysis.** The attack succeeds if the data contains a pair that satisfies the truncated differential characteristic of  $E_0$  (i.e., leads to a zero difference in at least one byte in the active column in round 0), and in addition, for one of the ‘friend pairs’ of that pair, the corresponding plaintext pair  $(P_3^j, P_4^j)$  has zero difference in either byte 10 or 15. With 64 plaintext pairs and 128 ‘friend pairs’ for each pair, each of these events occurs with probability of about  $1 - e^{-1} \approx 0.63$ , and hence, under standard randomness assumptions, the success probability of the attack is about  $0.63^2 \approx 0.4$ . This probability can be increased significantly by increasing the number of pairs we start with and the number of their ‘friend pairs’. For example, with 128 plaintext pairs and 128 friend pairs for each of them, the expected success probability is  $(1 - e^{-2})(1 - e^{-1}) \approx 0.54$ .

We note that the success probability can be increased further by exploiting other ways to cancel terms in Eq. (8). For example, if for some  $j, j'$ , the unordered pairs  $\{(P_3^j)_{10}, (P_4^j)_{10}\}$  and  $\{(P_3^{j'})_{10}, (P_4^{j'})_{10}\}$  are equal, then we can use the XOR of Eq. (8) for both pairs to cancel out the effect of subkey byte  $k_{-1,10}$  on the equation. This allows us to apply the efficient MITM attack described above also in cases where no ‘friend pair’ of  $(P_1, P_2)$  satisfies Eq. (10), thus increasing

the success probability of the attack. Our analysis shows that under standard randomness assumptions, for the same amount of 64 initial pairs and 128 ‘friend pairs’ for each pair considered above, this improvement increases the success probability of the attack from 0.4 to about 0.5.

The data complexity of the attack, for the success probability 0.4 computed above, is  $2 \cdot 2^6 \cdot 2^7 = 2^{14}$  chosen plaintexts and  $2^{14}$  adaptively chosen ciphertexts. We note that the amount of chosen plaintexts can be reduced by considering two structures of 8 plaintexts each (where in the first structure we have  $(P_1)_5 = 00_x$  and  $(P_1)_0$  assumes 8 different values, and in the second structure we have  $(P_2)_5 = 01_x$  and  $(P_2)_0$  assumes 8 different values) and taking the 64 pairs  $(P_1, P_2)$  composed of one plaintext in each structure. (In such a case, the ‘friend pairs’ are also taken in structures obtained by XORing the same value to all elements in the two initial structures.) This reduces the data complexity to slightly more than  $2^{14}$  adaptively chosen plaintexts and ciphertexts (as the number of encrypted plaintexts is negligible with respect to the number of decrypted ciphertexts). On the other hand, this slightly reduces the success probability of the attack, due to dependencies between the examined pairs  $(P_1, P_2)$ , as demonstrated in the next subsection. To conclude, with data complexity of  $2^{15}$  adaptively chosen plaintexts and ciphertexts we obtain success probability of more than 50%.

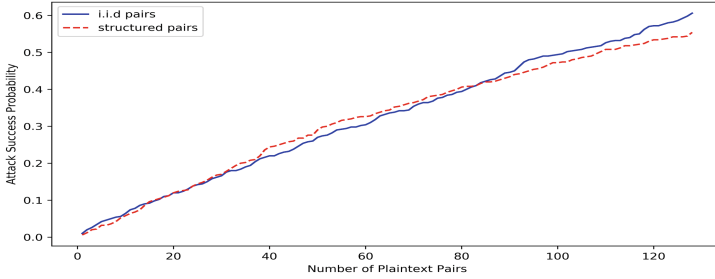
The memory complexity of the attack is no more than  $2^9$  128-bit memory cells, like in the yoyo attack of Rønjom et al. [30].

As for the time complexity, it is dominated by several steps that consist of about  $2^{16}$  simple operations each. The comparison of these operations to AES encryptions is problematic, and hence, we adopt a common strategy of counting the number of S-box applications and dividing it by 80, which is the number of S-boxes in 5-round AES. The number we obtain (divided by  $2^{16}$ ), in addition to the  $2^{14} + 2^{11}$  full encryptions of Step 4(b), is: negligible for Steps 1 and 4(c), 2 for Step 4(a), 6 for Step 4(d), 8 for Step 4(e), and  $24 \cdot 3 = 72$  for Step 5. Hence, the total complexity is less than  $2^{16.5}$  full encryptions.

We conclude that our 5-round attack requires  $2^{15}$  adaptively chosen plaintexts and ciphertexts,  $2^9$  memory and  $2^{16.5}$  time, and recovers the full secret key with success probability of more than 50%.

## 4.5 Experimental Verification

To verify the success probability of our attack computed above, we implemented two variants of the 5-round attack. The first variant uses up to 128 *independent* plaintext pairs. The second variant uses two structures, one of 8 plaintexts and another of 16 plaintexts, to create a total of 128 plaintext pairs. For each pair  $(P_1, P_2)$ , we generated 128 friend pairs. We ran the attack on 500 different randomly generated keys. For each success of the attack, we saved the number of pairs we had to try before finding the key. Then we extracted from this data the success probability of the attack, as a function of the amount of available data. Figure 6 shows this success probability, as a function of the number of plaintext pairs, up to a maximum of 128 pairs.



**Fig. 6.** Attack success probability

It can be seen that the success probability is slightly lower than the probability predicted by the above analysis. In particular, for 64 initial pairs, the success probability is slightly higher than 0.3 (rather than the predicted 0.4). We conjecture that the deviation from the theoretical estimate occurs due to dependency issues, but leave this small discrepancy for further research. Anyway, for data complexity of  $2^{15}$ , the experimental success probability is well above 50%.

The source code used in the experiments, along with the raw data, is included as a supplementary material, and will be made public together with the online version of the paper.

## 5 Improved Attack on 5-Round AES with a Secret S-Box

In [32], Tiessen et al. initiated the study of *AES with a secret S-box*, namely a variant of AES in which the SB operation is replaced by a key-dependent S-box. They showed that 5 rounds of the new variant can be broken with complexity of  $2^{40}$  and 6 rounds can be broken with complexity of  $2^{90}$ , using variants of the Square attack on AES [29]. In the last four years, six more papers analyzed 5-round variants of AES with a secret S-box: in [15, 25, 31] using the Square attack, in [24, 25] using impossible differentials, in [21] using impossible differentials and the multiple-of- $n$  property, and in [4] using the yoyo technique. The best currently known result was obtained by Bardeh and Rønjom [4] – data complexity of  $2^{32}$  adaptively chosen plaintexts and ciphertexts and time complexity of  $2^{31}$  operations (in addition to generating the data).

In this section we use the retracing boomerang technique to devise an attack on 5-round AES with a secret S-box with a complexity of  $2^{25.8}$  in the adaptively chosen plaintext and ciphertext model. Like the attacks of [4, 21, 24, 25, 31], our attack recovers the secret key, without fully recovering the secret S-box. (Actually, we recover the S-box up to an invertible affine transformation in  $(GF(2))^8$ ; as our attack is of a differential nature, it cannot distinguish between secret S-boxes that differ by such transformation.) On the other hand, it applies even against a stronger variant in which MC is also replaced by a key-dependent MDS transformation (see [16]) applied on each column. Among the previous attacks,

only the Square attack of Tiessen et al. [32] applies to this variant and can break it with complexity of  $2^{40}$ .

Our attack uses the same retracing boomerang framework as our attack on 5-round AES. Namely, we start with plaintext pairs  $(P_1, P_2)$  with difference only in bytes 0, 5, 10, 15, and for each such pair, we modify the corresponding ciphertext pair  $(C_1, C_2)$  into one of its mixture counterparts, which we denote by  $(C_3, C_4)$ , and ask for its decryption. We know that with probability  $2^{-6}$ , the corresponding pair  $(Z_3, Z_4)$  of intermediate values at the input of round 1 has zero difference in an inverse shifted column (e.g., in bytes 0, 5, 10, 15). (Note that this part does not use the specific structure of SB or of MC, and hence, can be applied also to a variant of AES with key-dependent SB and MC operations). Our goal now is to use this knowledge to attack round 0, as the attack we used for 5-round AES heavily relies on the fact that the S-box is known to the adversary.

**Partial recovery of the secret S-box.** To attack round 0, we use the strategy proposed in the structural attack of Biryukov and Shamir on SASAS [13], that was already used against AES with a secret S-box in [32], albeit inside the framework of the Square attack. Assume w.l.o.g. that the retracing boomerang predicts zero difference in byte 0 of the state  $Z$ , i.e., yields the equation  $(Z_3)_0 \oplus (Z_4)_0 = 0$ . (In the actual attack, if the procedure with byte 0 leads to a contradiction, the adversary has to perform it again with bytes 1, 2, 3.) By Eq. (6), we can rewrite this equation as

$$0 = (Z_3)_0 \oplus (Z_4)_0 = 02_x \cdot ((W_3)_0 \oplus (W_4)_0) \oplus 03_x \cdot ((W_3)_1 \oplus (W_4)_1) \oplus 01_x \cdot ((W_3)_2 \oplus (W_4)_2) \oplus 01_x \cdot ((W_3)_3 \oplus (W_4)_3). \quad (11)$$

Note that each of the values  $(W_3)_j$  has the form  $\text{SB}(P_3 \oplus k_{-1,j'})$ , where for  $j = 0, 1, 2, 3$ ,  $j' = \text{SR}^{-1}(j)$  takes the value 0, 5, 10, 15, respectively. Therefore, if we define  $4 \cdot 256 = 1024$  variables  $x_{m,j} = \text{SB}(m \oplus k_{-1,j'})$  (for  $m = 0, 1, \dots, 255$  and  $j' = 0, 1, 2, 3$ ), then each plaintext pair  $(P_1, P_2)$  for which the corresponding intermediate values  $(Z_3, Z_4)$  satisfy

$$(Z_3)_0 \oplus (Z_4)_0 = 0, \quad (12)$$

provides us with a linear equation in the variables  $\{x_{m,j}\}$ .

In order to recover the variables  $\{x_{m,j}\}$  by solving a system of linear equations, we need many pairs  $(Z_3, Z_4)$  that satisfy Eq. (12) simultaneously. We obtain these pairs by attaching about  $2^{10}$  ‘friend pairs’ to each original pair  $(P_1, P_2)$ , like we did in the attack on 5-round AES in Sect. 4. Hence, we start with  $2^6$  pairs  $(P_1, P_2)$ , and for each pair and about  $2^{10}$  friend pairs we perform the mixing retracing boomerang process and use each of the pairs to obtain a linear equation in the variables  $\{x_{m,j}\}$ . (This part of the attack has to be repeated for  $\ell = 0, 1, 2, 3$ , as each value of  $\ell$  leads to different equations. The equations presented above correspond to  $\ell = 0$ .) Then, we recover as many as we can of the variables  $\{x_{m,j}\}$  by solving a system of linear equations. We take a bit more than  $2^{10}$  friend pairs for each pair in order to obtain extra filtering, which allows us to efficiently discard pairs  $(P_1, P_2)$  that do not satisfy the boomerang property.

As was shown in [32], the equations do not allow determining the variables  $\{x_{m,j}\}$  (and thus, the secret S-box) completely. Indeed, as our basic Eq. (11) represents differences and not actual values, it is invariant under composition of the secret S-box with an invertible linear transformation over  $(GF(2))^8$ . Thus, the best we can obtain at this stage is four functions  $S_0, S_1, S_2, S_3$ , such that

$$S_j(x) = L_0(\text{SB}(x \oplus k_{-1,j'})),$$

for some unknown invertible linear transformation  $L_0$ . In addition, by repeating the attack for three other columns in round 0 (using the fact that for a pair  $(P_1, P_2)$  that satisfies the boomerang property, an entire inverse shifted column of  $Z_3 \oplus Z_4$  equals zero), we obtain the S-boxes  $S_j(x)$  for all  $j \in \{0, 1, \dots, 15\}$ , albeit with multiplication by a different matrix  $L_t$  in all the S-boxes of (inverse shifted) Column( $t$ ).

**Recovering the secret key.** While this information does not recover the S-box completely, it does allow us to recover the secret key  $k_{-1}$ , up to 256 possible values. This is done in two steps.

First, for each  $j' \in \{1, 2, 3\}$  we can easily recover  $\bar{k}_{j'} = k_{-1,0} \oplus k_{-1,j'}$  in time  $2^8$ , as  $\bar{k}_{j'}$  is the unique value of  $c$  such that  $S_j(x) = S_0(x \oplus c)$  for all  $x$ . In a similar way, we can recover each inverse shifted column of  $k_{-1}$  up to 256 possible values (e.g., to find the values  $k_{-1,1} \oplus k_{-1,s}$  for  $s \in \{6, 11, 12\}$  by attacking Column 3). This already reduces the number of possible values of  $k_{-1}$  to  $2^{32}$ .

Second, we find the differences  $k_{-1,0} \oplus k_{-1,j}$  for  $j = 1, 2, 3$  by taking several quartets of values  $(x_1, x_2, x_3, x_4)$  such that  $S_0(x_1) \oplus S_0(x_2) \oplus S_0(x_3) \oplus S_0(x_4) = 0$  and finding the unique value of  $c_j$  such that

$$S_j(c_j \oplus x_1) \oplus S_j(c_j \oplus x_2) \oplus S_j(c_j \oplus x_3) \oplus S_j(c_j \oplus x_4) = 0.$$

(The quartets are used to eliminate the effect of the difference between the linear transformations  $L_0$  and  $L_j$  in the definitions of  $S_0$  and  $S_j$ .) Thus, in about  $2^{12}$  operations we recover the entire secret key  $k_{-1}$ , up to the value of a single byte  $k_{-1,0}$ . Assuming that the secret S-boxes are determined by the secret key, the attack can be completed by exhaustive search over the  $2^8$  remaining key possibilities. The resulting attack algorithm is given in Algorithm 3.

**Attack analysis.** The data complexity of the attack is  $2^6 \cdot 2 \cdot 2^{10} = 2^{17}$  chosen plaintexts and  $2^{17}$  adaptively chosen ciphertexts. Like in the attack on 5-round AES presented in Sect. 4, we can reduce the required amount of chosen plaintexts to about  $2^{14}$  using structures, and so the overall data complexity is less than  $2^{17.5}$  adaptively chosen plaintexts and ciphertexts.

The time complexity is dominated by solving a system of 1034 equations in 1024 variables in Step 10, that has to be performed for each of the  $2^6$  pairs  $(P_1, P_2)$  and for  $\ell = 0, 1, 2, 3$ . Using the Four Russians Algorithm ([1]; see [3] for the motivation for choosing it), each solution of the system takes about  $(2^{10})^3 / \log(2^{10}) \approx 2^{27}$  simple operations, that are equivalent to about  $2^{27} / 80 \approx 2^{21}$  encryptions. Hence, the time complexity of the attack is  $2^{29}$ . (Note that the

**Algorithm 3.** Attack on 5-Round AES with Secret S-Box and MixColumns

- 
- 1: Ask for the encryption of  $2^6$  pairs  $(P_1, P_2)$  of chosen plaintexts that have non-zero difference only in bytes 0,5,10,15.
  - 2: **for all** Plaintext pairs  $(P_1, P_2)$  **do**
  - 3:     Generate  $2^{10} + 10$  ‘friend pairs’  $(P_1^j, P_2^j)$ , such that for each  $j$ :  $(P_1^j \oplus P_2^j) = P_1 \oplus P_2$ , and  $(P_1^j)_{\{0,5,10,15\}} = (P_1)_{\{0,5,10,15\}}$ .
  - 4:     Ask for the encryption of all ‘friend pairs’  $(P_1^j, P_2^j)$
  - 5: **for all** pairs  $(P_1, P_2)$  and for each  $\ell \in \{0, 1, 2, 3\}$  **do**     ▷ We present the case of  $\ell = 0$ , the other cases are similar.
  - 6:     **for all**  $m \in \{0, 1, \dots, 255\}$  and  $j \in \{0, 1, 2, 3\}$  **do**
  - 7:         Define  $x_{m,j} = SB(m \oplus k_{-1,SR^{-1}(j)})$
  - 8:         Assume that Eq. (11) is satisfied for all  $Z_3^j, Z_4^j$  of the ‘friend pairs’  $(P_1^j, P_2^j)$
  - 9:         Obtain the corresponding linear system of equations in  $x_{m,j}$
  - 10:         Solve the system of 1034 linear equations in 1024 variables
  - 11:         **if** a contradiction is reached **then**
  - 12:             Increment  $\ell$
  - 13:         **if**  $\ell > 3$  **then**
  - 14:             Discard the pair
  - 15:         **else**
  - 16:             The solution yields four functions  $S_j(x) = L_0(SB(x \oplus k_{-1,SR^{-1}(j)}))$ , for some unknown invertible linear transformation  $L_0$ .
  - 17:         Repeat the attack on the other three columns with  $(P_1, P_2)$  to obtain  $S_j(x)$  for  $j = 4, 5, \dots, 15$ .
  - 18: Find the rest of the secret key by exhaustive key search (assuming the secret S-box depends on the master 128-bit key  $k_{-1}$ )
- 

solution of a system of equations in Step 17 is much cheaper, as it has to be performed only for a single pair  $(P_1, P_2)$ .)

The memory complexity is dominated by the memory required for solving the system of equations, which is less than  $2^{17}$  128-bit blocks. (There is no need to store the plaintext/ciphertext pairs, as they can be analyzed ‘on the fly’.)

We conclude that the data complexity of the attack is  $2^{17.5}$  adaptively chosen plaintexts and ciphertexts, the time complexity is  $2^{29}$  encryptions, and the memory complexity is  $2^{17}$  128-bit blocks.

**Improving the overall complexity by applying a distinguisher before the attack.** Note that in the attack, we have to apply the equation-solving step  $2^8$  times, since we do not know which pair  $(P_1, P_2)$  and which value of  $\ell$  satisfies the boomerang property. Hence, if we can obtain this information in some other way, this will speedup the attack considerably.

A possible way to find a pair that satisfies the boomerang condition is to apply the yoyo distinguishing attack on 5-round AES of Rønjom et al. [30], which does not depend on knowledge of the S-box, and thus, can be applied in the secret S-box setting. (Note however that this attack depends on the MDS property of MC (see [16]). Hence, unlike the attack described above which applies when MC is replaced by an arbitrary invertible linear transformation, this attack



applies only if the transformation is assumed to satisfy the MDS property.) The attack of [30] requires  $2^{25.8}$  adaptively chosen plaintexts and ciphertexts, and in addition to distinguishing 5-round AES from a random permutation, it finds a pair  $(P_1, P_2)$  with non-zero difference only in bytes 0, 5, 10, 15, such that the corresponding intermediate values  $(Z_1, Z_2)$  have non-zero difference in only two bytes. This pair satisfies our boomerang property, and thus, can be used (along with 1034 friend pairs) in the attack described above. This reduces the complexity of each equation-solving step to  $2^{21}$ , and thus, the overall complexity of the attack is dominated by the complexity of Rønjom et al.'s attack. We conclude that this variant of the attack has data and time complexities of  $2^{25.8}$  and memory complexity of  $2^{17}$ .

## 6 The Retracing Rectangle Attack – Connection to Mixture Differentials

In this section we present the *retracing rectangle* attack, which is the retracing variant of the rectangle attack [6]. First we recall the amplified boomerang (a.k.a. rectangle) attack, then we present and analyze the new retracing rectangle attack, and then we use our new framework to expose a relation of the recently introduced *mixture differential* attack [22] to the rectangle attack.

### 6.1 The Amplified Boomerang (a.k.a. Rectangle) Attack

An apparent drawback of the boomerang attack is the need to use adaptively chosen plaintexts and ciphertexts – a very strong ability for the attacker. In [26], Kelsey et al. presented the amplified boomerang attack, which imitates the procedure of the boomerang attack using only chosen plaintexts. In the attack, the adversary considers *pairs of pairs* of plaintexts  $((P_1, P_2), (P_3, P_4))$  such that  $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ , and for each of them, she checks whether the corresponding quartet of ciphertexts  $((C_1, C_2), (C_3, C_4))$  satisfies  $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ . For the analysis of the attack, we refer the reader to [26].

Kelsey et al. applied the amplified boomerang attack to the AES' candidates MARS and SERPENT. In a subsequent work, Biham et al. [6] presented several enhancements of the attack, and gave it the name rectangle attack, which is the currently more commonly-used name.

### 6.2 The Retracing Rectangle Attack

The transformation from the retracing boomerang attack to the retracing rectangle attack is similar to the transformation from the (classical) boomerang attack to the rectangle attack.

**The attack setting.** We assume that  $E$  can be decomposed as  $E = E_1 \circ E_{02} \circ E_{01}$ , where  $E_{01}$  consists of dividing the state into two parts (a left part of  $b$  bits and a right part of  $n - b$  bits) and applying to them the functions  $E_{01}^L, E_{01}^R$ . Furthermore, we suppose that for  $E_{01}^L$ , there exists a differential characteristic  $\alpha_L \xrightarrow{p_1^L} \mu_L$ , for  $E_{01}^R$ , there exists a differential characteristic  $\alpha_R \xrightarrow{p_1^R} \mu_R$ , for  $E_{02}$ , there exists a differential characteristic  $\mu \xrightarrow{p_2} \beta$ , and for  $E_1$ , there exists a differential characteristic  $\gamma \xrightarrow{q} \delta$  (see Fig. 7).

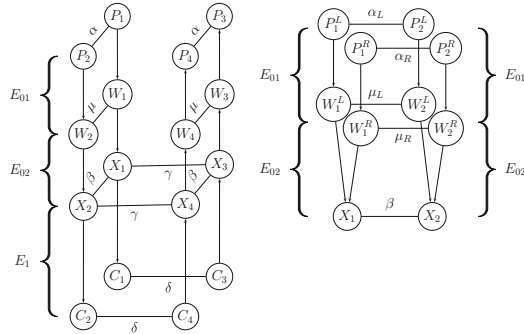


Fig. 7. The retracing rectangle setting

Assuming that  $p_1^R p_1^L p_2 q \gg 2^{-n/2}$ , the rectangle attack can be used to distinguish  $E$  from a random permutation, with data complexity of  $O((p_1^R p_1^L p_2 q)^{-1} \cdot 2^{n/2})$  chosen plaintexts. Recall that in the standard rectangle attack, we consider quartets of plaintexts  $((P_1, P_2), (P_3, P_4))$  such that  $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ , and check whether the corresponding quartets of ciphertexts  $((C_1, C_2), (C_3, C_4))$  satisfy  $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ . In the retracing rectangle attack, we consider only quartets of plaintexts that satisfy

$$(P_1 \oplus P_2 = \alpha) \wedge (P_3 \oplus P_4 = \alpha) \wedge ((P_1)^L \oplus (P_3)^L = 0 \text{ or } \alpha^L). \tag{13}$$

As a result, the two unordered pairs  $(P_1^L, P_2^L)$  and  $(P_3^L, P_4^L)$  are identical, and hence, if one of them satisfies the differential characteristic of  $E_{10}^L$ , then so does the other. Thus, the probability of the rectangle distinguisher is improved by a factor of  $(p_1^L)^{-1}$ .

**Advantages.** Unlike the shifting retracing boomerang attack, here we obtain an improvement in the probability of the distinguisher without a need to discard some part of the data. (This holds since the adversary can choose the plaintexts as she wishes, and in particular, can force the additional restriction  $(P_1^L \oplus P_3^L = 0 \text{ or } \alpha^L)$  ‘for free’.) In addition, the signal to noise ratio is improved, like in the retracing boomerang attack.

It should however be noted that in most applications of the rectangle attack, the adversary starts with structures  $S$  of pairs with input difference  $\alpha$ , such that each pair-of-pairs within the same structure satisfies the initial condition of the rectangle distinguisher. Then, for each structure, the adversary uses a hash table to check all these  $\binom{|S|}{2}$  quartets in time  $|S|$ . In the retracing rectangle attack, one has to either give up the structures and work with each pair-of-pairs that satisfies Eq. (13) separately, or else perform the ordinary rectangle attack and then check the additional condition ( $P_1^L \oplus P_3^L = 0$  or  $\alpha^L$ ) simultaneously with the condition  $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$  (which can be done using a hash table). In either case, the overall data complexity of the attack is not reduced, compared to the rectangle attack with structures, and thus, improvement of the signal to noise ratio is the main advantage of the retracing rectangle technique.

**A mixing variant – relation to mixture differentials.** Like in the mixing retracing boomerang attack, the adversary can force equality between the unordered pairs  $(P_1^L, P_2^L), (P_3^L, P_4^L)$  by choosing  $P_3 = (P_2^L, P_1^R)$  and  $P_4 = (P_1^L, P_2^R)$ , or in other words, by taking the pair  $(P_3, P_4)$  to be the *mixture counterpart* of the pair  $(P_1, P_2)$ . As this choice also forces equality between the pairs  $(P_1^R, P_2^R)$  and  $(P_3^R, P_4^R)$ , the probability of the rectangle distinguisher is increased by a factor of  $(p_1^L p_1^R)^{-1}$ .

Interestingly, it turns out that the core step of the *mixture differential* attack of Grassi [22] on 5-round AES fits into the mixture retracing rectangle attack framework.

Specifically, the core of [22]’s result is a chosen plaintext distinguishing attack on a 3.5-round variant of AES. In this attack, 3.5-round AES is decomposed as  $E_1 \circ E_{02} \circ E_{01}$ , where  $E_{01}$  consists of the first 1.5 rounds,  $E_{02}$  consists of a single MC layer, and  $E_1$  is composed of the last 1.5 rounds. The attack uses quartets of plaintexts  $(P_1, P_2, P_3, P_4)$  constructed by a mixing procedure, as described in Definition 1, and considers the corresponding quartets  $(X_1, X_2, X_3, X_4)$  and  $(Y_1, Y_2, Y_3, Y_4)$  of intermediate values after  $E_{01}$  and  $E_{02}$ , respectively. The representation of 1.5-round AES as four Super-S-boxes applied in parallel [16] allows deducing that  $X_1 \oplus X_2 \oplus X_3 \oplus X_4 = 0$  holds with probability 1. As  $E_{02}$  is linear, the same holds for  $Y_1, Y_2, Y_3, Y_4$ . Finally, the attack uses a truncated differential characteristic of  $E_1$  with probability 1 that starts with difference 0 in an inverse shifted column (e.g., bytes 0, 5, 10, 15) and ends with difference 0 in a shifted column (e.g., bytes 0, 7, 10, 13). (This characteristic also follows from the Super-S-boxes representation of 1.5-round AES.) If the pair  $(Y_1, Y_3)$  satisfies the input difference of this characteristic – an event that occurs with probability of  $2^{-32}$  – then  $(Y_2, Y_4)$  satisfies the input difference as well, and then we know for sure that both  $(C_1, C_3)$  and  $(C_2, C_4)$  have zero difference in bytes 0, 7, 10, 13. This provides a 64-bit filtering, that is exploited in [22] to obtain a key recovery attack on 5-round AES.

While this may not be apparent at a first glance, this attack is indeed a variant of the mixing retracing rectangle attack described above. The choice of plaintext quartets is exactly the same, and so is the treatment of  $E_1$  (taking note that the differential characteristics used in a boomerang/rectangle attack

may be truncated, as mentioned above). The only seeming difference is  $E_0$ , where instead of considering a specific differential characteristic we only make sure that the four outputs sum up to zero. However, this is actually the same as using all possible differential characteristics simultaneously, as is commonly done in boomerang/rectangle attacks.

## 7 Summary and Open Problems

In this paper we introduced a new version of boomerang attacks called a retracing boomerang attack, and used it to significantly improve the best known key recovery attacks on 5 rounds of AES (both in its standard form and when the S-box and the linear transformation are secret key-dependent components). The most interesting problems left open in this paper are:

- Find additional applications of the new technique.
- Find other types of correlations which can further increase the probability of the combined differential property.
- Create a “grand unified theory” of boomerang-like attacks which will explore their hidden relationships and treat them rigorously.

**Acknowledgements.** The authors thank the anonymous referees and Senyang Huang for their proposals and suggestions for improving the manuscript.

The research was supported in part by the European Research Council under the ERC starting grant agreement n. 757731 (LightCrypt), by the BIU Center for Research in Applied Cryptography and Cyber Security, by the Israel Ministry of Science and Technology, the Center for Cyber, Law, and Policy, by the Israel National Cyber Bureau in the Prime Minister’s Office, and by the Israeli Science Foundation through grants No. 3380/19, No. 880/18 and No. 1523/14.

The first author is a member of the Center for Cyber, Law, and Policy at the university of Haifa. The second author is a member of the BIU Center for Research in Applied Cryptography and Cyber Security. The third author is a member of CPIIS.

## References

1. Arlazarov, V., Dinic, E., Kronrod, A.M., Faradžev, I.: On economical construction of the transitive closure of a directed graph. *Dokl. Akad. Nauk SSSR* **194**(11), 1201–1290 (1970)
2. Bar-On, A., Dunkelman, O., Keller, N., Ronen, E., Shamir, A.: Improved key recovery attacks on reduced-round AES with practical data and memory complexities. In: Shacham, H., Boldyreva, A. (eds.) *CRYPTO 2018*. LNCS, vol. 10992, pp. 185–212. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-96881-0\\_7](https://doi.org/10.1007/978-3-319-96881-0_7)
3. Bard, G.V.: Achieving a  $\log(n)$  speed up for boolean matrix operations and calculating the complexity of the dense linear algebra step of algebraic stream cipher attacks and of integer factorization methods. *IACR Cryptology ePrint Archive*, 2006:163 (2006)
4. Bardeh, N.G., Rønjom, S.: Practical attacks on reduced-round AES. In: Buchmann, J., Nitaj, A., Rachidi, T. (eds.) *AFRICACRYPT 2019*. LNCS, vol. 11627, pp. 297–310. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-23696-0\\_15](https://doi.org/10.1007/978-3-030-23696-0_15)

5. Biham, E., Biryukov, A., Dunkelman, O., Richardson, E., Shamir, A.: Initial observations on Skipjack: cryptanalysis of Skipjack-3XOR. In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 362–375. Springer, Heidelberg (1999). [https://doi.org/10.1007/3-540-48892-8\\_27](https://doi.org/10.1007/3-540-48892-8_27)
6. Biham, E., Dunkelman, O., Keller, N.: The rectangle attack — rectangling the Serpent. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 340–357. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-44987-6\\_21](https://doi.org/10.1007/3-540-44987-6_21)
7. Biham, E., Keller, N.: Cryptanalysis of Reduced Variants of Rijndael (1999). Unpublished manuscript
8. Biham, E., Perle, S.: Conditional linear cryptanalysis - cryptanalysis of DES with less than  $2^{42}$  complexity. IACR Trans. Symmetric Cryptol. **2018**(3), 215–264 (2018)
9. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. J. Cryptol. **4**(1), 3–72 (1991). <https://doi.org/10.1007/BF00630563>
10. Biryukov, A., De Cannière, C., Dellkrantz, G.: Cryptanalysis of SAFER++. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 195–211. Springer, Heidelberg (2003). [https://doi.org/10.1007/978-3-540-45146-4\\_12](https://doi.org/10.1007/978-3-540-45146-4_12)
11. Biryukov, A., Khovratovich, D.: Related-key cryptanalysis of the full AES-192 and AES-256. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 1–18. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-10366-7\\_1](https://doi.org/10.1007/978-3-642-10366-7_1)
12. Biryukov, A., Leurent, G., Perrin, L.: Cryptanalysis of Feistel networks with secret round functions. In: Dunkelman, O., Keliher, L. (eds.) SAC 2015. LNCS, vol. 9566, pp. 102–121. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-31301-6\\_6](https://doi.org/10.1007/978-3-319-31301-6_6)
13. Biryukov, A., Shamir, A.: Structural cryptanalysis of SASAS. J. Cryptol. **23**(4), 505–518 (2010). <https://doi.org/10.1007/s00145-010-9062-1>
14. Cid, C., Huang, T., Peyrin, T., Sasaki, Y., Song, L.: Boomerang connectivity table: a new cryptanalysis tool. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018. LNCS, vol. 10821, pp. 683–714. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-78375-8\\_22](https://doi.org/10.1007/978-3-319-78375-8_22)
15. Cui, T., Chen, H., Mesnager, S., Sun, L., Wang, M.: Statistical integral distinguisher with multi-structure and its application on AES-like ciphers. Cryptogr. Commun. **10**(5), 755–776 (2018). <https://doi.org/10.1007/s12095-018-0286-5>
16. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography. Springer, Heidelberg (2002). <https://doi.org/10.1007/978-3-662-04722-4>
17. Dinur, I., Dunkelman, O., Keller, N., Shamir, A.: Efficient dissection of composite problems, with applications to cryptanalysis, knapsacks, and combinatorial search problems. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 719–740. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-32009-5\\_42](https://doi.org/10.1007/978-3-642-32009-5_42)
18. Dunkelman, O., Keller, N.: Treatment of the initial value in time-memory-data tradeoff attacks on stream ciphers. Inf. Process. Lett. **107**(5), 133–137 (2008)
19. Dunkelman, O., Keller, N., Shamir, A.: A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony. J. Cryptol. **27**(4), 824–849 (2013). <https://doi.org/10.1007/s00145-013-9154-9>
20. Ferguson, N., et al.: Improved cryptanalysis of Rijndael. In: Goos, G., Hartmanis, J., van Leeuwen, J., Schneier, B. (eds.) FSE 2000. LNCS, vol. 1978, pp. 213–230. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-44706-7\\_15](https://doi.org/10.1007/3-540-44706-7_15)
21. Grassi, L.: MixColumns properties and attacks on (round-reduced) AES with a single secret S-box. In: Smart, N.P. (ed.) CT-RSA 2018. LNCS, vol. 10808, pp. 243–263. Springer, Cham (2018). [https://doi.org/10.1007/978-3-319-76953-0\\_13](https://doi.org/10.1007/978-3-319-76953-0_13)

22. Grassi, L.: Mixture differential cryptanalysis: a new approach to distinguishers and attacks on round-reduced AES. *IACR Trans. Symmetric Cryptol.* **2018**(2), 133–160 (2018)
23. Grassi, L.: Probabilistic mixture differential cryptanalysis on round-reduced AES. In: Paterson, K.G., Stebila, D. (eds.) *SAC 2019*. LNCS, vol. 11959, pp. 53–84. Springer, Cham (2020). [https://doi.org/10.1007/978-3-030-38471-5\\_3](https://doi.org/10.1007/978-3-030-38471-5_3)
24. Grassi, L., Rechberger, C., Rønjom, S.: Subspace trail cryptanalysis and its applications to AES. *IACR Trans. Symmetric Cryptol.* **2016**(2), 192–225 (2016)
25. Hu, K., Cui, T., Gao, C., Wang, M.: Towards key-dependent integral and impossible differential distinguishers on 5-round AES. In: Cid, C., Jacobson Jr., M. (eds.) *SAC 2018*. LNCS, vol. 11349, pp. 139–162. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-10970-7\\_7](https://doi.org/10.1007/978-3-030-10970-7_7)
26. Kelsey, J., Kohno, T., Schneier, B.: Amplified boomerang attacks against reduced-round MARS and Serpent. In: Goos, G., Hartmanis, J., van Leeuwen, J., Schneier, B. (eds.) *FSE 2000*. LNCS, vol. 1978, pp. 75–93. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-44706-7\\_6](https://doi.org/10.1007/3-540-44706-7_6)
27. Murphy, S.: The return of the cryptographic boomerang. *IEEE Trans. Inf. Theory* **57**(4), 2517–2521 (2011)
28. US National Bureau of Standards: Data Encryption Standard, Federal Information Processing Standards publications no. 46 (1977)
29. US National Institute of Standards and Technology: Advanced Encryption Standard, Federal Information Processing Standards publications no. 197 (2001)
30. Rønjom, S., Bardeh, N.G., Helleseeth, T.: Yoyo tricks with AES. In: Takagi, T., Peyrin, T. (eds.) *ASIACRYPT 2017*. LNCS, vol. 10624, pp. 217–243. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-70694-8\\_8](https://doi.org/10.1007/978-3-319-70694-8_8)
31. Sun, B., Liu, M., Guo, J., Qu, L., Rijmen, V.: New insights on AES-like SPN ciphers. In: Robshaw, M., Katz, J. (eds.) *CRYPTO 2016*. LNCS, vol. 9814, pp. 605–624. Springer, Heidelberg (2016). [https://doi.org/10.1007/978-3-662-53018-4\\_22](https://doi.org/10.1007/978-3-662-53018-4_22)
32. Tiessen, T., Knudsen, L.R., Kölbl, S., Lauridsen, M.M.: Security of the AES with a secret S-box. In: Leander, G. (ed.) *FSE 2015*. LNCS, vol. 9054, pp. 175–189. Springer, Heidelberg (2015). [https://doi.org/10.1007/978-3-662-48116-5\\_9](https://doi.org/10.1007/978-3-662-48116-5_9)
33. Tunstall, M.: Improved “Partial Sums”-based square attack on AES. In: *SECURITY 2012*, pp. 25–34 (2012)
34. Wagner, D.: The boomerang attack. In: Knudsen, L. (ed.) *FSE 1999*. LNCS, vol. 1636, pp. 156–170. Springer, Heidelberg (1999). [https://doi.org/10.1007/3-540-48519-8\\_12](https://doi.org/10.1007/3-540-48519-8_12)