Improved Boomerang Attacks on 6-Round AES

Augustin Bariant[1,2], Orr Dunkelman[3], Nathan Keller[4], Gaëtan Leurent[2], and Victor Mollimard[3]

[1] ANSSI, Paris, France
augustin.bariant@ssi.gouv.fr
[2] Inria, Paris, France
gaetan.leurent@inria.fr
[3] Computer Science Department, University of Haifa, Haifa, Israel
orrd@cs.haifa.ac.il, victor.mollimard@gmail.com
[4] Department of Mathematics, Bar Ilan University, Ramat Gan, Israel
Nathan.Keller@biu.ac.il

Abstract. The boomerang attack is a cryptanalytic technique which allows combining two short high-probability differentials into a distinguisher for a large number of rounds. Since its introduction by Wagner in 1999, it has been applied to many ciphers. One of the best-studied targets is a 6-round variant of AES, on which the boomerang attack is outperformed only by the dedicated Square attack. Recently, two new variants of the boomerang attack were presented: retracing boomerang (Eurocrypt'20) and truncated boomerang (Eurocrypt'23). These variants seem incompatible: the former achieves lower memory complexity by throwing away most of the data in order to force dependencies, while the latter achieves lower time complexity by using large structures, which inevitably leads to a large memory complexity.
In this paper we show that elements of the two techniques can be combined to get 'the best of the two worlds' – the practical memory complexity of the retracing attack and the lower time complexity of the truncated attack. We obtain an attack with data complexity of 2^{57} (compared to 2^{59} and 2^{55} of truncated and retracing boomerang, respectively), memory complexity of 2^{33} (compared to 2^{59} and 2^{31}), and time complexity of 2^{61} (compared to 2^{61} and 2^{80}). This is the second-best attack on 6-round AES, after the Square attack.

Keywords: AES · Boomerang attack

1 Introduction

Differential attacks, introduced by Biham and Shamir [5] in 1990, use the evolution of differences through the encryption process to attack block ciphers and other cryptographic primitives. In order to make ciphers immune to differential attacks, various design strategies were developed, which allow proving that any 'long' differential (i.e. differential which covers many rounds of the cipher) has a very low probability. One of these strategies is the 'wide trail strategy' [10] deployed in the AES [27] – the most widely used cipher today. This strategy allows proving that any differential which covers 4 of the 10 rounds of AES has probability of less than 2^{-113} [21], making it very unlikely that a differential attack will be a threat for the security of the full (10-round) AES.

In 1999, Wagner [28] developed the boomerang technique which allows bypassing this barrier, as it enables combining two 'short' differentials which cover r_1 and r_2 rounds of the cipher into a 'long' property of $r_1 + r_2$ rounds. Assume that the differential of the first r_1 rounds of the cipher is of the form $\alpha \to \beta$ (i.e. difference α goes to difference β) with probability p, and that the differential for the subsequent r_2 rounds is of the form $\gamma \to \delta$ with probability q. The boomerang attack on $r_1 + r_2$ rounds considers a pair of plaintexts with difference α, looks at the corresponding ciphertext pair, XORes δ to the two ciphertexts, decrypts the new ciphertexts and checks whether the resulting plaintexts have a difference α. An easy analysis shows that under standard independence assumptions, the resulting difference is equal to α with probability p^2q^2. This allows attacking ciphers which have no high-probability 'long' differentials, provided that they have high-probability short differentials.

In the 25 years since its introduction, the boomerang technique has been used to attack numerous ciphers. One of its major successes was attacking variants of the AES – the best known attacks on the full AES in the related-key model [7,11], as well as the best known attacks on 5-round AES in the standard (single-key) model [14] are boomerang attacks. A specific variant of AES which has been especially well-studied is 6-round AES. Four different types of boomerang attacks were applied to it in the last 20 years – by Biryukov [6], Dunkelman *et al.* [14], Rahman *et al.* [25], and Bariant and Leurent [3]. This variant is especially important as it is the largest variant on which practical-complexity attacks are known, and also as it is used as a component in several other cryptosystems – e.g., TNT-AES [1], AES-PRF-192 [23], and WEM [8]. It should be mentioned that on 6-round AES, the boomerang technique does not provide the best known attack, as it is outperformed by the dedicated Square attack [12,16].

Over the years, many variants and enhancements of the boomerang technique were proposed. Let us focus on two recently proposed variants.

At Eurocrypt 2020, Dunkelman *et al.* [14] presented the *retracing boomerang* – a variant of the boomerang technique in which before XORing δ to the ciphertexts, the adversary discards most of the ciphertext pairs, leaving only pairs which satisfy a certain condition. The condition ensures that the second differential holds for sure in the backward direction, thus increasing the overall probability of the boomerang to p^2q. On the other hand, the extra condition disallows using *structures* in the attack, which eventually leads to a larger time complexity. Dunkelman *et al.* applied their technique to 6-round AES, on which the best previous boomerang attack, by Biryukov [6], had data and time complexity of 2^{71}. Dunkelman *et al.* managed to reduce the data complexity to 2^{55} adaptively chosen plaintexts and ciphertexts (ACPC), at the expense of increasing the time complexity to 2^{80} encryptions. The retracing boomerang attack was used also to attack other variants of AES, including 5-round AES and AES with a secret S-box.

At Eurocrypt 2023, Bariant and Leurent [3] presented the *truncated boomerang* attack – a variant of the boomerang technique in which a truncated differential is used instead of a 'standard' one. The main feature of the technique is the extensive use of *structures*, both on the plaintext side and on the ciphertext side where an entire structure of ciphertexts is XORed with many possible values. Another central feature is embedding the key recovery parts into the boomerang distinguisher. Bariant and Leurent applied their technique to 6-round AES and managed to reduce the time complexity of the attack to 2^{61} encryptions, while keeping the data complexity not so high, at 2^{59} ACPC. On the other hand, due to the use of large structures, the memory complexity of the attack of [3] was much higher than the memory complexity of the previous attacks – 2^{59} 128-bit blocks instead of 2^{33} in [6] and 2^{31} in [14]. The truncated boomerang attack was used also to attack other variants of AES, as well as other AES-based ciphers including Kiasu-BC [19], Deoxys-BC [20], and TNT-AES [1].

The retracing boomerang attack and the truncated boomerang attack seem inherently incompatible – while the former cannot use structures due to the extra condition on the ciphertexts, the main gain of the latter stems from using large ciphertext structures. Nevertheless, we show that elements of the two attacks can be combined to get 'the best of the two worlds' – the practical memory complexity of the retracing boomerang attack along with the lower time complexity of the truncated boomerang attack, while keeping the data complexity roughly unchanged. We present two main attacks. The first has a low data complexity of 2^{51} ACPC (compared to 2^{59} in [3] and 2^{55} in [14]), a memory complexity of 2^{32} 128-bit blocks (compared to 2^{59} in [3] and 2^{31} in [14]), and a time complexity of 2^{68} (compared to 2^{61} in [3] and 2^{80} in [14]). We also propose a time/memory trade-off for this attack. The second attack has a lower time complexity, of around 2^{61} encryptions, competing with the truncated boomerang attack. However, compared to the first attack, its data complexity is increased to 2^{57} ACPC, and its memory complexity to 2^{33}. This is nevertheless the second-best attack on 6-round AES, after the Square attack, in terms of overall complexity. We hope that like the retracing boomerang attack and the truncated boomerang attack, the new combination will lead to improved attacks on other variants of AES and on other cryptosystems that use reduced-round AES as a component. A comparison of our results with previous attacks on 6-round AES is presented in Table 1.

This paper is organized as follows. In Section 2 we describe the structure of the AES. In Section 3 we present background on the boomerang attack and its variants. In Section 4 we present our new attacks on 6-round AES. In Section 5 we present a 'distinguisher' of 6-round AES which seems better than all previously known distinguishers on this variant, but which actually fails due to a subtle incompatibility issue. We conclude the paper in Section 6.

Rounds	Type	Data	Time	Mem	Ref
6	Imp. Diff.	2^{76}	2^{104}	2^{45}	[30]
6	Prob. Mixture Diff.	2^{73}	2^{105}	2^{33}	[17]
6	Mixture diff.	2^{31}	2^{73}	2^{31}	[2]
6	Mixture diff.	2^{44}	2^{63}	2^{44}	[29]
6	Square	2^{35}	2^{45}	2^{32}	[16]
6	Square	2^{33}	2^{40}	2^{32}	[12]
6	Boomeyong	2^{80}	2^{78}	2^{28}	[25]
6	Boomerang	2^{71}	2^{71}	2^{33}	[6]
6	Retracing Boomerang	2^{55}	2^{80}	2^{31}	[14]
6	Truncated Boomerang	2^{59}	2^{61}	2^{59}	[3]
6	Boomerang	2^{51}	2^{68}	2^{32}	subsection 4.1
6	Boomerang	2^{51}	2^{66}	2^{42}	subsection 4.1
6	Boomerang	2^{57}	2^{61}	2^{33}	subsection 4.2

Table 1. Comparison of our attacks with selected previous key-recovery attacks on 6-round AES.

2 AES

AES [10] (originally named Rijndael) is a 128-bit block cipher developed by Daemen and Rijmen in 1997. In 2001, the US National Institute of Science and Technology decided to standardize it under the name *Advanced Encryption Standard (AES)* and from then, it became one of the most used block ciphers worldwide, if not the most used one.

AES is a classical Substitution-Permutation Network transforming a 128-bit state organized as a 4×4 array of 8-bit words. The encryption process repeats 10, 12 or 14 times (respectively for key of 128-bit, 192-bit and 256-bit) a round function composed itself by four sub-functions as illustrated in Figure 1:

SubBytes. Apply a known 8-bit S-box independently to the bytes of the state;
ShiftRows. Shift each row of the state to the left by the position of the row;
MixColumns. Multiply each column by the same known invertible 4-by-4 matrix over the finite field $GF(2^8)$;
AddRoundKey. Add a 128-bit round key computed from the secret key to the state, using a bitwise XOR operation.

An additional AddRoundKey operation is performed before the first round and the last MixColumns operation is omitted. As properties of the key schedule are not used in this paper, we refer readers to [27] for its description. In this paper we consider a reduced version with 6 rounds, and we assume that the last MixColumns operation is omitted for simplicity.

Notation r is the number of rounds and the rounds are numbered from 0 to $r-1$. At round ℓ, x_ℓ, y_ℓ, z_ℓ and w_ℓ denote the states before respectively the SubBytes, ShiftRows, MixColumns and AddRoundKey operations. The initial subkey is denoted k_0 and the subkey added during the AddRoundKey

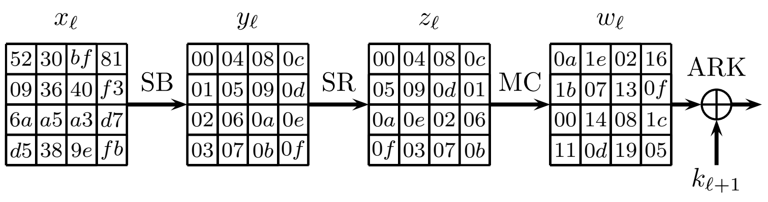

Fig. 1. A round of AES.

operation of round ℓ is denoted $k_{\ell+1}$. Given a 16-byte state (or subkey) X, $X[4j+i]$ denotes the byte in column j and row i of X, as shown on Figure 2. A set of bytes $X[i], X[j], X[l]$ is denoted $X[i,j,l]$. In the rest of the paper, the XOR between state values is represented with the addition (+). To indicate byte (resp. column/diagonal/anti-diagonal) indexes in some states of the trail, we use letters $a \in \{0, \dots 15\}$ and $\ell, m \in \{0,1,2,3\}$. Bytes are indexed with the conventional order presented in Figure 2. Diagonals and anti-diagonals are indexed with the index of their active column in the first row. We denote $\tilde{a} = -a \mod 4, \tilde{\ell} = -\ell \mod 4, \tilde{m} = -m \mod 4$.

00	04	08	12
01	05	09	13
02	06	10	14
03	07	11	15

Fig. 2. Byte numbering of a 16-byte AES state or subkey.

3 Background on the boomerang attack

3.1 Differentials

When considering a binary function $E : \mathbb{F}_2^n \to \mathbb{F}_2^n$, a differential is a pair of input, output differences $(\Delta_{\text{in}}, \Delta_{\text{out}}) \in \{0,1\}^{2n}$. A differential is said to exist if there exists a message m such that $E(m) + E(m + \Delta_{\text{in}}) = \Delta_{\text{out}}$. It is said to have probability $p = \mathbb{P}\left[E(m) + E(m + \Delta_{\text{in}}) = \Delta_{\text{out}}\right]$ defined over a random plaintext m. As common in the literature we assume that the probability itself does not depend on the key (but rather which pairs satisfy the differential). If the key impacts the probability too much, one may obtain weak-key classes.

3.2 The basic boomerang attack

In 1999, Wagner [28] proposed to split the encryption function E into two halves $E = E_1 \circ E_0$ for each of which a high probability differential exists denoted respectively $\Delta_{\text{in}} \xrightarrow{p} \Delta_{\text{out}}$ of probability p for E_0 and $\nabla_{\text{in}} \xrightarrow{q} \nabla_{\text{out}}$ of probability q for E_1. Wagner proposed to use these two differentials by constructing a quartet $(P, P', \bar{P}, \bar{P}')$ of the following form:

$$\big(P,\ P+\Delta_{\text{in}},\ E^{-1}[E(P)+\nabla_{\text{out}}],\ E^{-1}[E(P+\Delta_{\text{in}})+\nabla_{\text{out}}]\big) \tag{1}$$

To simplify the explanation and corresponding to Figure 3, we write $C = E(P)$, and $E_0(P) = I = E_1^{-1}(C)$ and extend the notation to $P', \bar{P}, \bar{P}'$ to $C', \bar{C}, \bar{C}'$ and $I', \bar{I}, \bar{I}'$. In the case of the AES, we also extend the internal state notations x_ℓ, x'_ℓ, $\bar{x}_\ell$, $\bar{x}'_\ell$ (and similarly for y, z, and w).

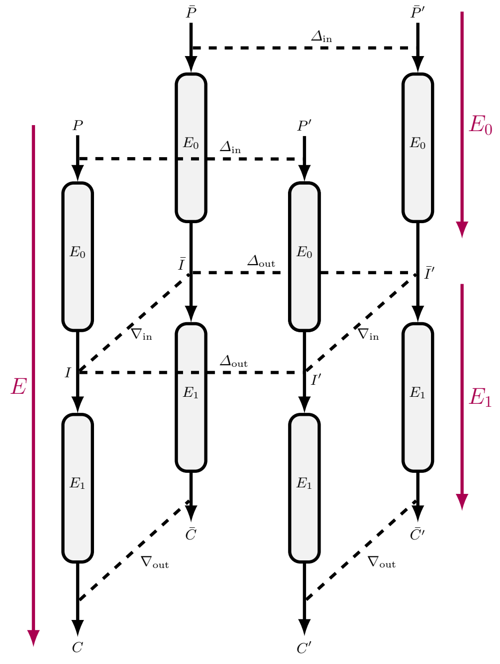

Fig. 3. Illustration of the basic boomerang quartet adapted from [18].

Wagner remarked that the probability p_b that a quartet $(P, P', \bar{P}, \bar{P}')$ verifies $\bar{P} + \bar{P}' = \Delta_{\text{in}}$ was greater for E than for a random function when p^2q^2 was significantly greater than 2^{-n} (and assuming that E_0 and E_1 could be considered independent). Indeed, we can write:

$$
\begin{aligned}
p_b &= \mathbb{P}\left[\bar{P} + \bar{P}' = \Delta_{\text{in}}\right] \qquad (2)\\
&\geq \mathbb{P}\left[E_0^{-1}(\bar{I}) + E_0^{-1}(\bar{I}') = \Delta_{\text{in}} | \bar{I} + \bar{I}' = \Delta_{\text{out}}\right] \cdot \mathbb{P}\left[\bar{I} + \bar{I}' = \Delta_{\text{out}}\right]\\
&\geq p \cdot \mathbb{P}\left[\bar{I} + \bar{I}' = \Delta_{\text{out}} | I + I' = \Delta_{\text{out}}\right] \cdot \mathbb{P}\left[I + I' = \Delta_{\text{out}}\right]\\
&\geq p \cdot \mathbb{P}\left[\bar{I} + \bar{I}' = I + I'\right] \cdot \mathbb{P}\left[E_0(P) + E_0(P') = \Delta_{\text{out}}\right]\\
&\geq p^2 \cdot \mathbb{P}\left[I + \bar{I} = \nabla_{\text{in}}\right] \cdot \mathbb{P}\left[I' + \bar{I}' = \nabla_{\text{in}}\right]\\
&\geq p^2 \cdot \mathbb{P}\left[E_1^{-1}(C) + E_1^{-1}(\bar{C}) = \nabla_{\text{in}}\right] \cdot \mathbb{P}\left[E_1^{-1}(C') + E_1^{-1}(\bar{C}') = \nabla_{\text{in}}\right]\\
&\geq p^2 q^2
\end{aligned}
$$

which can be used as a distinguisher when p^2q^2 is significantly greater than 2^{-n}, the expected probability of the constructed quartet $(P, P', \bar{P}, \bar{P}')$ to verify the boomerang property for a random function. Generally, this distinguisher can be extended in a key-recovery attack by leveraging its key dependencies.

The analysis of the boomerang attack has seen multiple improvements. Boomerangs with multiple differentials were analysed by Wagner [28] in his original paper. Plaintext-only variants, the amplified boomerang [22] and the rectangle [4] attacks, were presented shortly after. Multiple works were then dedicated to the analysis of the boomerang switch, i.e. the middle rounds of the boomerang attack [15,24,9]. Among those, Murphy showed in 2011 that some boomerang characteristics were in fact impossible [24]. However, we will not dive into the analysis of the boomerang switch, since our attacks exploit a 'free' boomerang switch, as highlighted by the retracing boomerang attack [14], which we examine in the next section.

3.3 Retracing boomerang on AES

The retracing boomerang attack is a framework presented by Dunkelman *et al.* at Eurocrypt 2020 [14]. To this day, the framework is the fastest key-recovery attack against 5-round AES, with or without secret S-boxes, and produces interesting attacks on 6-round AES as well. Our attacks compare to their key-recovery attack on 6-round AES, presented in [13, Appendix C.3].

Core idea. The retracing boomerang makes use of correlated values in the ciphertext side such that the independence assumption between the equations of Equation 2 does not hold. Instead, the lower differential $\nabla_{\text{out}} \to \nabla_{\text{in}}$ through E_1^{-1} happens with probability q for both $(C, \bar{C})$ and $(C', \bar{C}')$ at the same time. The events are therefore correlated positively and this increases the boomerang probability to $p_b = p^2q$.

The main idea of the retracing boomerang on AES is to define ∇_{out} active on a single anti-diagonal, such that the returning pairs $(C, C{+}\nabla_{\text{out}})$ and $(C', C'{+}\nabla_{\text{out}})$ have the same pair of values on the active anti-diagonal: *i.e.* either $C = C'$ or $C = C' + \nabla_{\text{out}}$ on the active anti-diagonal.

Dunkelman *et al.* propose two types of retracing boomerangs in their framework, as highlighted by Figure 4:

- The mixing retracing boomerang: chose ∇_{out} active on an anti-diagonal, such that $\nabla_{\text{out}} = C + C'$ on the active anti-diagonal. The mixing retracing boomerang is very close to the yoyo attack of [26]. In that case, $C = \bar{C}'$ and $C' = \bar{C}$ on the anti-diagonal.
- The shifting retracing boomerang: only keep pairs (C, C') such that $C = C'$ on an anti-diagonal, and take any ∇_{out} active only on the corresponding state word. In that case, $C = C'$ and $\bar{C} = \bar{C}'$ on the anti-diagonal.

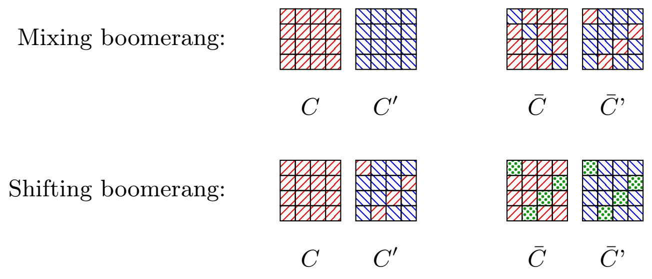

Fig. 4. Mixing and shifting for retracing boomerangs.

Key-recovery attack on 6-round AES. A key-recovery attack on 6-round AES is presented in the full version of the Retracing Boomerang framework [13, Appendix C.3], and is a shifting retracing boomerang attack. Since our results are heavily inspired by this attack, we explain this attack in details, using the characteristics from Figure 5. Algorithm 1 describes the attack.

Characteristics analysis. Starting from a pair (P, P') with an active diagonal, we consider that it is a *right forward pair* if it follows the forward characteristic of Figure 5. We require that one byte of the first column (in position ℓ) is inactive at the end of the first round (probability 2^{-6}), and that the main diagonal is inactive at the end of the third round (probability 2^{-32}). Therefore a forward pair (P, P') is right with probability 2^{-38}.

In the backward direction, $\bar{C}$ and $\bar{C}'$ are obtained by shifting C and C' with a difference in the main antidiagonal. We assume that the difference collapses to only the first byte of the state after one round (state z_4), and that the difference is the same for the pairs $(C, \bar{C})$ and $(C', \bar{C}')$. This happens with probability $2^{-2\times 24-8} = 2^{-56}$ with random ciphertexts, but the attacker will guess the value of key bytes $k_6[0, 7, 10, 13]$ in order to directly construct values $\bar{C}$ and $\bar{C}'$ with this property.

Starting from z_4, we have a shifting retracing boomerang: C and C' have the same value on the main antidiagonal, and $\bar{C}$ and $\bar{C}'$ differ from C and C' by the same difference in a single byte, included in the main antidiagonal. Thefore the active S-box transition from z_4 to x_4 is the same for the pairs $(C, \bar{C})$ and $(C', \bar{C}')$: $x_4 + \bar{x}_4 = x'_4 + \bar{x}'_4$.

The transitions from z_3 to x_3 are also the same for both pairs because of the S-box switch: every active S-box j in the backward characteristics ($z_3[j] + \bar{z}_3[j] = z'_3[j] + \bar{z}'_3[j] \neq 0$) is inactive in the forward characteristic ($z_3[j] = z'_3[j]$). Therefore, we have $\bar{z}_3[j] = \bar{z}'_3[j]$, and $S^{-1}(z_3[j]) + S^{-1}(\bar{z}_3[j]) = S^{-1}(z'_3[j]) + S^{-1}(\bar{z}'_3[j])$. Therefore, we obtain $x_3 + \bar{x}_3 = x'_3 + \bar{x}'_3$ and $z_2 + \bar{z}_2 = z'_2 + \bar{z}'_2$ with probability one.

When considering the pair $(\bar{C}, \bar{C}')$, we deduce that the difference in z_2 is the same as in the forward pair: $\bar{z}_2 + \bar{z}'_2 = z_2 + z'_2$. In particular, antidiagonal $\tilde{\ell}$ is inactive in z_2; this implies that diagonal $\tilde{\ell}$ of w_0 is inactive.

To summarize: assuming that (P, P') is a right forward pair, and that $\bar{C}, \bar{C}'$ are constructed such that $z_4 + \bar{z}_4 = z'_4 + \bar{z}'_4$ with this difference only active in byte 0, then with probability 1 a diagonal of w_0 is inactive for the pair $\bar{C}, \bar{C}'$ (the same diagonal that is inactive for (P, P')).

Attack description. The attack proceeds as follows, with 8 fixed values $\delta_0, \ldots \delta_7$ active only on byte 0:

1. Ask for the encryption of a structure of 2^{20} plaintexts with different values on the main diagonal.
2. For each candidate K for $k_6[0, 7, 10, 13]$, partially decrypt the ciphertexts to compute $Y = MC^{-1}(x_5)$, define new ciphertexts $\bar{C}_i$ such that $\bar{Y}_i = Y + \delta_i$ and query and store their corresponding plaintexts $\bar{P}_i$.
3. For each candidate K for $k_6[0, 7, 10, 13]$, filter all pairs (P, P') such that $Y[0] = Y'[0]$.
   For each such pair, consider the 8 quartets $(Y, Y', \bar{Y}_i, \bar{Y}'_i)$ with $\bar{Y}_i = Y + \delta_i$ and $\bar{Y}'_i = Y' + \delta_i$, whose corresponding plaintexts $\bar{P}_i, \bar{P}'_i$ were queried during step 2.
   For each $\ell \in \{0, 1, 2, 3\}$, assume that the $\tilde{\ell}$-th diagonal of w_0 is inactive for all quartets simultaneously and deduce the key k_0. If a candidate k_0 is compatible with all quartets, return it as the correct key.

Complexity. With 2^{20} plaintexts, the expected number of right forward pairs is $2^{20} \cdot 2^{19} \cdot 2^{-38} = 2$. If there is a right forward pair (P, P'), then it satisfies $Y[0] = Y'[0]$. At step 3, with the correct key guess, all the quartets $(Y, Y', \bar{Y}_i, \bar{Y}'_i)$ follow the boomerang, hence w_0 has an inactive diagonal with probability 1. Therefore the correct key candidate will be recovered and the attack succeeds with high probability.

Reciprocally, a wrong guess of $k_0[0, 5, 10, 15] || k_6[0, 7, 10, 13]$ and ℓ passes the test with probability $2^{-8\times8}$; we expect on average $2^{39-8+32+32+2-64} = 2^{33}$ false positives. False positives are detected and discarded by recovering key candidates for another diagonal of k_0.

Algorithm 1 Retracing boomerang attack on 6-round AES.

```
Query the encryption of $2^{20}$ plaintexts with different values on the main diagonal
for all $k_6[0,7,10,13]$ do
    for all ciphertext $C$ do
        Partially decrypt $C$ to obtain $Y[0,1,2,3]$
        for $0 \leq i < 8$ do
            Define $\bar{Y}_i = Y + \delta_i, \bar{Y}_i' = Y' + \delta_i$, compute corresponding $\bar{C}_i, \bar{C}_i'$
            Query $\bar{P}_i = E^{-1}(\bar{C}_i), \bar{P}_i' = E^{-1}(\bar{C}_i')$
    for all pairs $(C, C')$ with $Y[0] = Y'[0]$ do
        for $0 \leq \ell < 4$ do
            Assume $w_0[\ell]$ is inactive for all quartets $(Y, Y', \bar{Y}_i, \bar{Y}_i')$
            Deduce key candidates for $k_0$
```

Step 3 iterates over 2^{32} keys, $2^{39} \cdot 2^{-8} = 2^{31}$ pairs for each key, and 4 values of ℓ. The complexity to recover the key candidates is estimated as equivalent to 2^{15} encryptions in [13].[5] Therefore the total time complexity is $2^{32+31+2+15} = 2^{80}$.

The data complexity is $2^{32} \cdot 2^{20} \cdot 8 = 2^{55}$ at step 2, and the memory complexity is $2^{20} \cdot 8$ to store the $\bar{P}_i$ and $\bar{P}_i'$ (step 2), resulting in:

$$(D, T, M) = (2^{55}, 2^{80}, 2^{23})$$

We observe that the 2^{55} decryption queries actually correspond to only $2^{20} \cdot 2^{32} = 2^{52}$ distinct ciphertexts. Therefore we can reduce the data complexity by storing all the queries:

$$(D, T, M) = (2^{52}, 2^{80}, 2^{52})$$

3.4 Truncated boomerang attack on AES

The truncated boomerang framework was presented in EUROCRYPT 2023 by Bariant *et al.* [3] to attack 6-round AES and other AES-based ciphers. The idea of the framework is to only consider truncated differential characteristics, for the upper and the lower parts of the cipher. This attack brings two main novelties. First, this allows to build big structures on the plaintext and the ciphertext sides, increasing the number of quartets for a given number of oracle queries. Second, instead of first fixing a distinguisher and adding key-recovery rounds before and after it, the truncated boomerang considers a boomerang trail on the full cipher, and recovers key candidates from each quartet, from the hypothesis that the quartet passes the boomerang trail (similarly to the 3R attacks defined on DES in [5]). Namely, each quartet is analyzed and suggests key material according

[5] The key-recovery procedure follows a Meet-in-The-Middle approach similar to our paper, but is slightly more conservative; We instead compute a complexity of $2^{13.7}$ encryptions, detailed in Section 4.1.

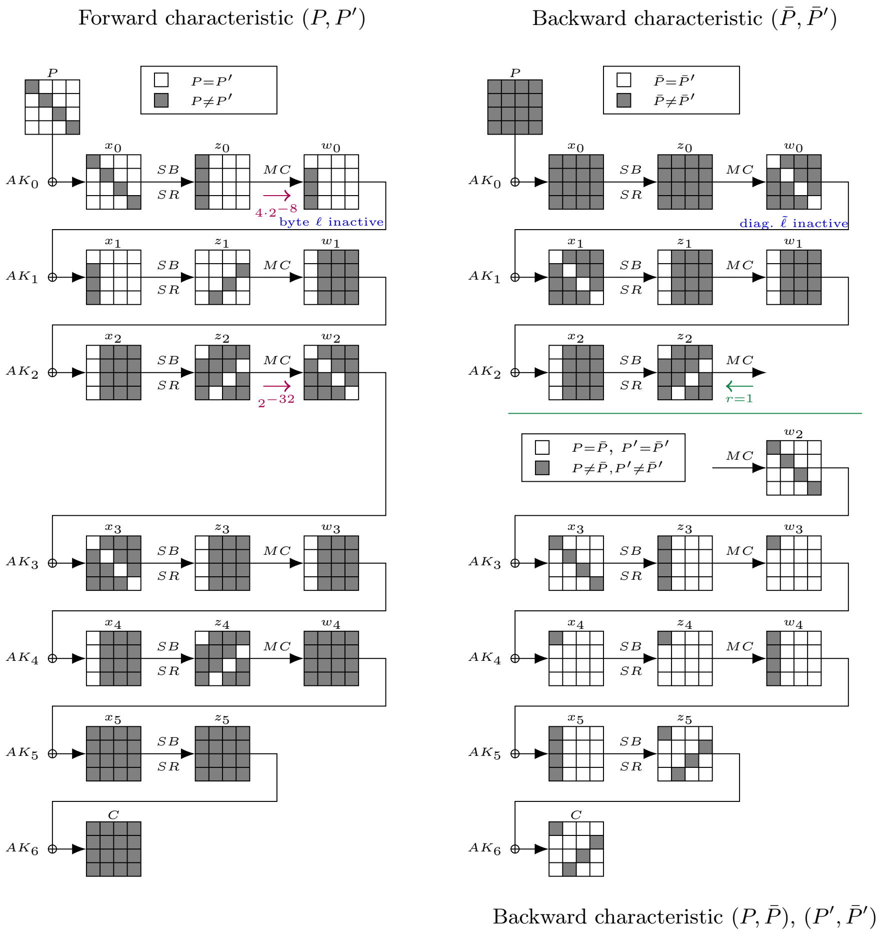
Forward characteristic (P, P')
Backward characteristic $(\bar{P}, \bar{P}')$
P
$P=P'$
$P\neq P'$
$\bar{P}=\bar{P}'$
$\bar{P}\neq\bar{P}'$
x_0 z_0 w_0
x_1 z_1 w_1
x_2 z_2 w_2
x_3 z_3 w_3
x_4 z_4 w_4
x_5 z_5
C
AK_0 AK_1 AK_2 AK_3 AK_4 AK_5 AK_6
SB SR MC
$4\cdot 2^{-8}$
byte ℓ inactive
2^{-32}
diag. $\tilde{\ell}$ inactive
$r=1$
$P=\bar{P},\ P'=\bar{P}'$
$P\neq\bar{P}, P'\neq\bar{P}'$
Backward characteristic $(P, \bar{P})$, $(P', \bar{P}')$

Fig. 5. Retracing boomerang key-recovery attack on 6-round AES.

to the differential transitions in the first and last rounds. The key suggested the most frequently is expected to be the correct one. This strategy saves on guessing parts of the last subkey, but rather allows to deduce it from quartets. The drawback of this attack is the high memory complexity needed to store quartet structures. This approach leads to a 6-round key-recovery attack with complexity:

$$(D, T, M) = (2^{59}, 2^{69}, 2^{59})$$

4 New attacks

The idea of our new attacks is to improve the retracing boomerang attack with different boomerang characteristics and new key-recovery strategies, following insights from the truncated boomerang attack.

These attacks are inspired by an observation regarding the 6-round retracing boomerang attack : the attack requires much more decryption queries (2^{55}) than encryption queries (2^{20}). This implies that we may discard some ciphertext pairs (C, C') at the cost of additional encryptions without increasing the overall time or data complexity. With that idea in mind, we filter ciphertext pairs and only consider pairs (C, C') which collide on two anti-diagonals of ciphertext, rather than a single byte at the end of the fifth round. For each ciphertext collision, we can then build a structure of 2^{32} shifted ciphertexts $C + \nabla_i$ (∇_i takes all possible values in one of the inactive anti-diagonals), following the shifting retracing boomerang framework. Note that unlike in the truncated boomerang attack, we do not build a structure for each ciphertext, but only for ciphertexts colliding on two inactive anti-diagonals. The plaintext structures and ciphertext structures, each of size 2^{32}, are such that asking a collision on 64 bits of ciphertext does not bring any significant complexity overhead. Indeed, with 2^{32} plaintext queries, we expect roughly 1 pair colliding on 64 bits, leading to 2^{32} decryption queries, and 2^{32} potential quartets. On the other hand, given a collision on two anti-diagonals of ciphertext, the pair (P, P') is more likely to be a *right forward pair.*

Compared to the retracing boomerang attack, we additionally use truncated trails in the backward trail of $(C, \bar{C})$ and $(C', \bar{C}')$ without fixing the value of δ_i, which improves the probability of the boomerang characteristic. It can be seen as a shifting retracing boomerang on 6 rounds, rather than on 5 rounds as in [13].

In the following attacks, the time complexity bottleneck comes from the key-recovery step. We therefore present slightly different boomerang characteristics and key-recovery strategies, leading to different trade-offs between time, data and memory complexities. Our results are summarized in Table 1.

4.1 A Key-Recovery Attack Optimized for Data Complexity

The first key-recovery attack that we propose is based on the boomerang depicted in Figure 6, and described in Algorithm 2.

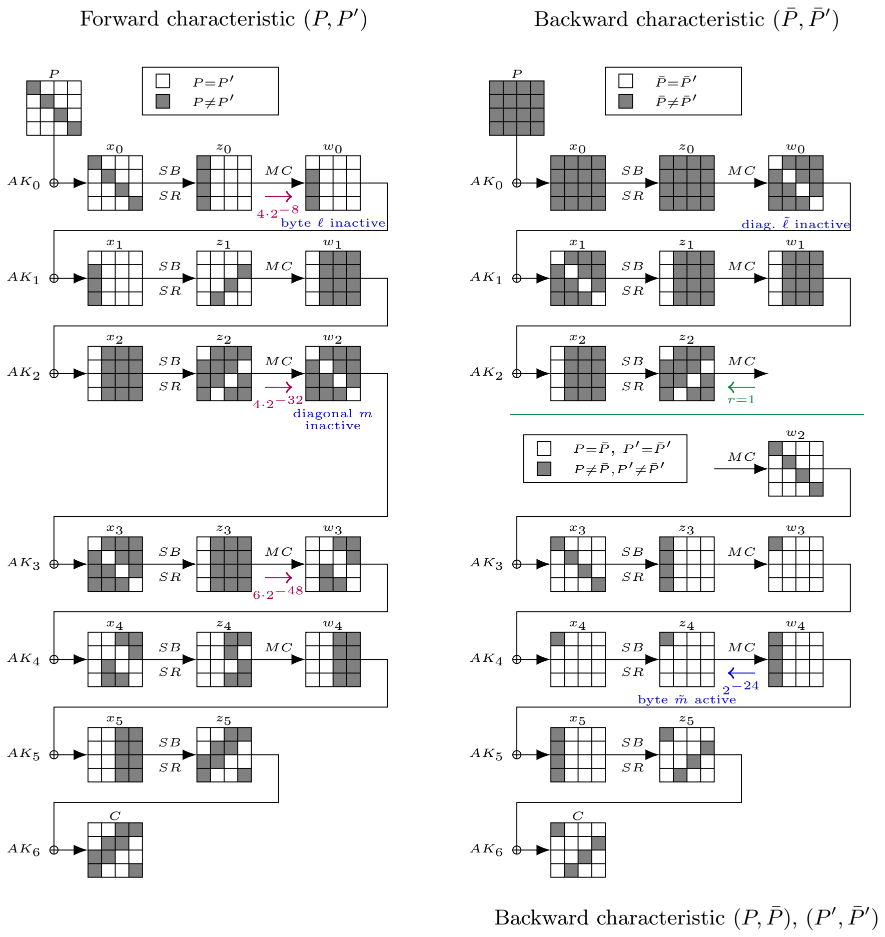

Fig. 6. Boomerang characteristic on 6-round AES with low data complexity.

Boomerang analysis. Starting from a pair (P, P') with an active diagonal, we consider that it is a *right forward pair* if one byte in position $\ell \in \{0, 1, 2, 3\}$ is inactive at the end of the first round (probability $4 \cdot 2^{-8}$), one diagonal in position $m \in \{0, 1, 2, 3\}$ is inactive at the end of the third round (probability $4 \cdot 2^{-32}$), and two diagonals are inactive at the end of the fourth round (probability $6 \cdot 2^{-48}$). Therefore a forward pair (P, P') is right with probability $2^{-81.4}$. This implies a collision in two anti-diagonals of ciphertext. The probability that a random ciphertext pair collides on two anti-diagonals is $6 \cdot 2^{-64} = 2^{-61.4}$.

In the backward direction, $\bar{C}$ and $\bar{C}'$ are obtained by shifting C and C' with a difference in one of the inactive anti-diagonals. This corresponds to a shifting retracing boomerang with a single active Super-box. Therefore, the active S-box transitions from z_5 to x_5 and from z_4 to x_4 are the same for the pairs $(C, \bar{C})$ and $(C', \bar{C}')$. We assume that the difference collapses to a single byte in position $\tilde{m}$ of the state z_4 (after one round), corresponding to the inactive diagonal m in $w_2 + w_2'$; this happens with probability 2^{-24}. In practice, the attacker will guess the value of key bytes $k_6[0, 7, 10, 13]$ in order to directly construct values $\bar{C}$ and $\bar{C}'$ with this property.

As in the retracing boomerang attack, we have $(z_3 + z_3')[0, 1, 2, 3] = 0$ and $(\bar{z_3} + \bar{z_3}')[0, 1, 2, 3] = 0$, which in turn implies $(x_3 + x_3')[0, 5, 10, 15] = 0$ and $(\bar{x}_3 + \bar{x}_3')[0, 5, 10, 15] = 0$. Therefore, we obtain $x_3 + \bar{x}_3 = x_3' + \bar{x}_3'$ and $z_2 + \bar{z}_2 = z_2' + \bar{z}_2'$ with probability one. When considering the pair $(\bar{C}, \bar{C}')$, we deduce that the difference in z_2 is the same as in the forward pair: $\bar{z}_2 + \bar{z}_2' = z_2 + z_2'$. In particular, anti-diagonal $\tilde{\ell}$ is inactive in z_2; this implies that diagonal $\tilde{\ell}$ of w_0 is inactive.

We then proceed as in the retracing boomerang attack. We obtain many quartets following the characteristic simultaneously: assuming that (P, P') is a right forward pair, and that the pairs $(\bar{C}, \bar{C}')$ are constructed such that $z_4 + \bar{z}_4$ is active only in byte $\tilde{m}$, then with probability 1, diagonal $\tilde{\ell}$ of w_0 is inactive for each pair $(\bar{C}, \bar{C}')$.

Comparison with the retracing boomerang attack. In total, the probability that a pair (P, P') is a right pair, and that a shifted pair $(\bar{C}, \bar{C}')$ follows the returning trail is $2^{-81.4-24} = 2^{-105.4}$. This is much lower than the 6-round AES retracing boomerang attack, that has a boomerang probability of $2^{-38} \times 2^{-24} = 2^{-62}$, if we intepret the lower characteristic as a truncated trail instead of a key guess. However, the probability for a wrong forward pair to have the right ciphertext difference is $2^{-61.4}$ (compared to 2^{-8} for the retracing boomerang, corresponding to the event $z_4[0] + \bar{z}_4[0] = 0$). In order to collect a single right quartet, there are $2^{62-8} = 2^{54}$ wrong quartets in the retracing boomerang, but only $2^{105.4-61.4} = 2^{44}$ wrong quartets remain in our attack. This gain in signal-to-noise ratio allows to significantly reduce the time complexity of the key-recovery, since the key-recovery procedure needs to be applied to fewer quartets.

Attack description. The base of the attack consists of the following steps:

1. Ask for the encryption of structures of 2^{32} plaintexts with different values on the main diagonal.

2. Among each structure, look for pairs of ciphertexts (C, C') colliding on any two anti-diagonals (for simplicity, we consider them as the first two). Let us denote $\bar{C}_i = C + \nabla_i$ and $\bar{C}'_i = C' + \nabla_i$ for $1 \leq i \leq 2^{32}$, where ∇_i takes the value i on the first anti-diagonal and the value 0 on other anti-diagonals. Ask for the decryptions $\bar{P}_i = E^{-1}(\bar{C}_i)$ and $\bar{P}'_i = E^{-1}(\bar{C}'_i)$ for all $1 \leq i \leq 16 \times 2^{24} = 2^{28}$.
3. For each candidate K for $k_6[0, 7, 10, 13]$, and each $\tilde{m} \in \{0, 1, 2, 3\}$, fetch 8 pairs $(P, \bar{P}_i)$ (already queried) such that under the key K, the first column of z_4 of $(P, \bar{P}_i)$ is active only in position $\tilde{m}$, and consider the 8 corresponding quartets $(P, P', \bar{P}_i, \bar{P}'_i)$.
   For each $\tilde{\ell} \in \{0, 1, 2, 3\}$, assume that diagonal $\tilde{\ell}$ of w_0 is inactive for all quartets simultaneously and deduce key k_0. If a candidate k_0 is compatible with all quartets, return it as the correct key.

Algorithm 2 Low data complexity boomerang attack on 6-round AES.

```
loop                                          ▷ The expected number of iterations is $2^{18.4}$
    Query the encryption of a structure of $2^{32}$ plaintexts
    for all pairs $(C, C')$ colliding on two anti-diagonals do
        for $0 \leq i < 2^{28}$ do
            Define $\bar{C}_i = C + \nabla_i$, $\bar{C}'_i = C' + \nabla_i$,
            Query $\bar{P}_i = E^{-1}(\bar{C}_i)$, $\bar{P}'_i = E^{-1}(\bar{C}'_i)$
        for all $k_6[0, 7, 10, 13]$ do
            for $0 \leq \tilde{m} < 4$ do
                Gather 8 quartets $(P, P', \bar{P}_i, \bar{P}'_i)$ s.t. $z_4 + \bar{z}_4$ is active only on position $\tilde{m}$
                for $0 \leq \tilde{\ell} < 4$ do
                    Assume diagonal $\tilde{\ell}$ of $w_0$ is inactive for $(P, P')$ and all $(\bar{P}_i, \bar{P}'_i)$
                    Deduce key candidates for $k_0$
```

Complexity. The attack queries new structures of plaintexts until a structure includes a right forward pair. Since the probability of the forward characteristic is $2^{-81.4}$, we expect $2^{32+31-81.4} = 2^{-18.4}$ right pairs per structure, and we expect to iterate over $2^{18.4}$ structures on average before finding a right pair. This corresponds to $2^{18.4+32+31-61.4} = 2^{20}$ candidates (C, C') at step 2. When a right pair is present, it passes the filtering of step 2, and the attack succeeds for the correct guess of $k_6[0, 7, 10, 13]$, $\tilde{\ell}$ and $\tilde{m}$.

We now give an analysis of Step 3. We first show that we can fetch the pairs $(P, \bar{P}_i)$ efficiently. Then, we explain in details the most technical part: the recovery of k_0 candidates from friend quartets, which we slightly improve compared to the meet-in-the-middle technique from [14].

Fetching the quartets. In Step 3, given a candidate K for $k_6[0, 7, 10, 13]$ and a position $\tilde{m} \in \{0, 1, 2, 3\}$, we need to fetch 8 pairs $(\bar{P}_i, \bar{P}'_i)$ (leading to 8 quartets $(P, P', \bar{P}_i, \bar{P}'_i)$) such that the state z_4 of pair $(P, \bar{P}_i)$ is active only in position $\tilde{m}$

of the active column under the key K. This can be done efficiently: start from $C = E(P)$, compute the first column of z_4 from the anti-diagonal of C, add the 256 possible byte values at position $\tilde{m}$ of the first column of z_4, and compute the corresponding values in the ciphertext side. This gives 256 different values for the first anti-diagonal in the ciphertexts. On average, this corresponds to $256 \times \frac{2^{28}}{2^{32}} = 16$ ciphertexts $\bar{C}_i$ of which the decryption was queried during Step 2. The probability that at least 8 such ciphertexts were queried during Step 2 can be approximated by $\Pr(\text{Poisson}(16) \geq 8) \geq 0.99$. The ciphertext $\bar{C}'_i$ is defined as $\bar{C}'_i = C + C' + \bar{C}_i$. With $256 \times 4 \times 2 = 2^{11}$ S-box calls, we fetch the 8 pairs; This is negligible compared to the complexity of getting the candidates for $k_0[0, 5, 10, 15]$ as explained below.

Meet-In-The-Middle procedure to recover key candidates. We start by examining two candidate quartets, and extracting on average 2^8 candidates $k_0[0, 5, 10, 15]$ for each $\tilde{\ell}$. Indeed, given two quartets $(P, P', \bar{P}_i, \bar{P}'_i)$ and $(P, P', \bar{P}_j, \bar{P}'_j)$, we have three pairs to use for filtering key candidates (P, P'), $(\bar{P}_i, \bar{P}'_i)$, $(\bar{P}_j, \bar{P}'_j)$, and each pair provides an 8-bit filtering.

Following [14], we use a meet-in-the-middle procedure, considering independently 2^{16} values of $k_0[0, 5]$ and 2^{16} values of $k_0[10, 15]$.

1. Create 8 tables $T_0, T_1, T_2, T_3, T'_0, T'_1, T'_2, T'_3$.
2. For all 2^{16} values of $k_0[0, 5]$, compute state $z_0[0, 1]$ of $P + P'$ (denoted $a[0, 1]$), $\bar{P}_i + \bar{P}'_i$ (denoted $b[0, 1]$), and $\bar{P}_j + \bar{P}'_j$ (denoted $c[0, 1]$). For each ℓ, store the 24-bit value
$$\mathsf{MC}(a[0], a[1], 0, 0)[\ell] \parallel \mathsf{MC}(b[0], b[1], 0, 0)[\ell] \parallel \mathsf{MC}(c[0], c[1], 0, 0)[\ell]$$
in table T_ℓ.
3. For all 2^{16} values of $k_0[10, 15]$, compute state $z_0[2, 3]$ of $P + P'$ (denoted $a[2, 3]$), $\bar{P}_i + \bar{P}'_i$ (denoted $b[2, 3]$), and $\bar{P}_j + \bar{P}'_j$ (denoted $c[2, 3]$). For each ℓ, store the 24-bit value
$$\mathsf{MC}(0, 0, a[2], a[3])[\ell] \parallel \mathsf{MC}(0, 0, b[2], b[3])[\ell] \parallel \mathsf{MC}(0, 0, c[2], c[3])[\ell]$$
in table T'_ℓ.
4. Look for collisions between T_ℓ and T'_ℓ, for $\ell \in \{0, 1, 2, 3\}$ (or equivalently for $\tilde{\ell} \in \{0, 1, 2, 3\}$).

Steps 2 and 3 each require roughly the computation of $3 \times 4 \times 2^{16}$ AES S-boxes. However, since the same pair (P, P') is used for multiple quartets (with different key guesses for k_6), we can precompute the values $\mathsf{MC}(a[0], a[1], 0, 0)$ for all keys $k_0[0, 5]$, and $\mathsf{MC}(0, 0, a[2], a[3])$ for all keys $k_0[10, 15]$, reducing the complexity to $2 \times 4 \times 2^{16} = 2^{19}$ AES S-boxes. Step 4. requires 2^{18} sequential lookups to match the lists, equivalent to 2^{18} AES S-boxes. In total we obtain a complexity equivalent to $2^{20} + 2^{18} = 2^{20.3}$ AES S-boxes, or equivalently $2^{13.7}$ 6-round AES encryptions, to recover 2^{10} candidates for $k_0[0, 5, 10, 15]$ and $\tilde{\ell}$.

Then, we can further filter the candidates using the remaining quartets. Indeed, each quartet provides an 8-bit filtering, and testing a key candidate requires only the evaluation of 2 AES columns (this adds negligible terms to the complexity). Given a wrong candidate pair (C, C'), the probability that there exists a choice of $\tilde{m}, \tilde{\ell}, k_6[0,7,10,13], k_0[0,5,10,15]$ compatible with 8 quartets is $2^{2+2+32+32-9\times 8} = 2^{-4}$. For the remaining candidates we apply the same procedure, to recover the other diagonals of k_0; this eliminates wrong candidates, and returns the full key k_0 for the right forward pair.

Finally, we obtain an attack with time complexity equivalent to $2^{20} \times 2^{32} \times 4 \times 2^{20.3} = 2^{74.3}$ AES S-boxes, or $2^{67.7}$ 6-round AES encryptions.

The attack requires on average $2^{32+18.4} = 2^{50.4}$ encryptions and $2 \times 2^{20+28} = 2^{49}$ decryptions, for a total of $2^{50.9}$ queries.

The memory complexity is bounded by the storage of each structure, i.e. 2^{32} 128-bit states. Therefore we obtain:

$$(D, T, M) = (2^{50.9}, 2^{67.7}, 2^{32})$$

A time-memory trade-off. The time complexity of this attack can be slightly decreased by remarking that in the attack, we apply the MiTM procedure multiple times to each quartet $(P, P', \bar{P}_i, \bar{P}'_i)$. Indeed, given a pair (P, P') with two colliding anti-diagonals in output, we perform 4×2^{32} MiTM procedures on 2^{28} existing quartets (for each activity position m in z_4, and each candidate for $k_6[0,7,10,13]$). Instead, it is possible to precompute 2^{18} candidates for $k_0[0,5,10,15]$ induced by each quartet with one MiTM procedure (with a variant using a single quartet) for each the 2^{28} quartets.

Step 3 is performed by recovering the list of 2^{18} candidates for $k_0[0,5,10,15]$ corresponding to two different quartets, and intersecting them. Therefore the time complexity becomes 2 table lookups recovering 2^{18} candidates each, and a matching step. It is hard to compare the cost of memory accesses to AES rounds, but since we make a large number of sequential accesses, we approximate it as equivalent to an S-box computation, resulting in a complexity of $2 \times 2^{20+2+32+18}$ S-box evaluations, or $2^{66.4}$ AES evaluations.

The memory complexity is however increased. Naively, we store 2^{28} sets of 2^{18} candidates, but it is possible to instead store only the sets of candidates for 2^{26} quartets: on average, 4 quartets among the 2^{26} chosen quartets correspond to each candidate $k_6[0,7,10,13]$ and activity position $\tilde{m}$ in z_4, and the MiTM procedure only needs to be applied to 2 of them (the 6 other quartets are used afterwards to filter the remaining candidates). Given $k_6[0,7,10,13]$ and $\tilde{m}$, the probability that at least 2 corresponding quartets are in the set of chosen quartets can be approximated to $\Pr(\text{Poisson}(4) \geq 2) = 0.91$; if this is not the case, with probability 0.09, we discard the candidate $(k_6[0,7,10,13], \tilde{m})$. This increases the expected number of plaintext structures to consider by a factor 1/0.91, but does not affect the time complexity because we do not process candidates with fewer than 2 precomputed quartets. This gives a memory complexity of $2^{26+18} = 2^{44}$ 32-bit states, equivalent to 2^{42} 128-bit states. In total, this gives:

$$(D, T, M) = (2^{51}, 2^{66.4}, 2^{42})$$

4.2 A Key-Recovery Attack Optimized for Time Complexity

This attack reuses the boomerang of subsection 4.1 with a slight change: the input pair (P, P') and returning pair $(\bar{P}, \bar{P}')$ are required to be inactive in a byte, in position $a \in \{0, 5, 10, 15\}$, i.e. in the main diagonal. This reduces the size of plaintext structures compared to the previous attack, but makes the key-recovery procedure more time-efficient. Indeed, for pairs following this characteristic, the first-round transitions depend on three key bytes ($k_0[5, 10, 15]$ if $a = 0$) rather than four. The attack is depicted in Figure 7 and described in Algorithm 3..

Boomerang analysis. Similarly to the previous attack, the probability that a pair is a *right forward pair* is $2^{-81.4}$, while the random probability that two ciphertexts collide in two anti-diagonals is $2^{-61.4}$.

In the backward direction, we assume that the difference collapses to a single byte in anti-diagonal $\tilde{m}$ after one round (state z_4); this happens with probability 2^{-24}. This implies that diagonal $\tilde{\ell}$ of w_0 is inactive for the pair $(\bar{C}, \bar{C}')$, with the same analysis as in the previous attacks. Moreover, we require $\bar{P} + \bar{P}'$ to be inactive in byte a; this happens with probability 2^{-8}.

As in the previous attacks, we consider multiple quartets, but the additional quartets are not required to collide in byte a of $\bar{P} + \bar{P}'$. Therefore, we use the same property as previously: assuming that (P, P') is a right forward pair, and that $(\bar{C}, \bar{C}')$ are constructed such that $z_4 + \bar{z}_4$ is active only in anti-diagonal m, then with probability 1, diagonal $\tilde{\ell}$ of w_0 is inactive for the pair $(\bar{C}, \bar{C}')$.

Comparison with the first attack. The boomerang probability is $2^{-81.4-24-8} = 2^{-113.4}$ compared to $2^{-105.4}$ in the first attack, and the probability that a wrong quartet is considered decreases by a factor 2^8 (corresponding to the condition $\bar{P}[0] = \bar{P}'[0]$). Therefore, this increases the data complexity but not the key-recovery time complexity, since the signal-to-noise ratio, corresponding to the number of quartets on which the key-recovery is performed, is unchanged. Additionally, the structure size is reduced by the same factor 2^8 ; more plaintext pairs are needed to find a right forward pair. This also increases the data complexity, but not the key-recovery time complexity, since this does not affect the signal-to-noise ratio. On the other hand, the key-recovery time complexity for a single quartet is decreased, because fewer candidates for bytes of k_0 need to be looped upon.

Attack description.

1. Ask for the encryption of structures of 2^{32} plaintexts with different values on the main diagonal.

2. Among each structure, consider pairs (P, P') with an inactive byte in the main diagonal of input; denote the inactive byte position as a. Look for corresponding pairs of ciphertexts (C, C') colliding on two anti-diagonals (for simplicity we consider them as the first two). Let us denote $\bar{C}_i = C + \nabla_i$ and $\bar{C}'_i = C' + \nabla_i$ for $1 \leq i \leq 2^{32}$, where ∇_i takes the value i on the first anti-diagonal and the value 0 on other anti-diagonals. Query and store the decryptions $\bar{P}_i = E^{-1}(\bar{C}_i)$ and $\bar{P}'_i = E^{-1}(\bar{C}'_i)$ for all $1 \leq i \leq 2^{32}$.
   For each $\ell \in \{0, 1, 2, 3\}$, assume that byte ℓ of w_0 is inactive for the pair (P, P'). Compute the set S_ℓ of 2^{16} candidates for $k_0[5, 10, 15]$ suggested by (P, P') under this assumption.
3. Consider plaintext pairs $(\bar{P}_i, \bar{P}'_i)$ that collide on byte a (we consider $a = 0$ in the rest of our analysis).
   For each $\ell \in \{0, 1, 2, 3\}$, consider the 2^{16} candidates for $k_0[5, 10, 15]$ in S_ℓ, and verify whether byte ℓ of w_0 is inactive for the pair $(\bar{P}_i, \bar{P}'_i)$. On average, 2^8 candidates should remain for each ℓ.
   Assume that state z_4 of $(P, \bar{P}_i)$ is active on byte $\tilde{m} \in \{0, 1, 2, 3\}$ of the first column and deduce 2^{10} candidates for $k_6[0, 7, 10, 13] \parallel \tilde{m}$. In total, a quartet suggests 2^{10} candidates for $k_0[5, 10, 15] \parallel \ell$ and 2^{10} candidates for $k_6[0, 7, 10, 13] \parallel \tilde{m}$ (2^{20} candidates in total).
4. For each quartet and each of its 2^{10} candidates for $k_6[0, 7, 10, 13] \parallel \tilde{m}$, consider 7 other pairs $(\bar{C}_j, \bar{C}'_j)$ such that the first column of z_4 of $(C, \bar{C}_j)$ is active only in position m; fetch their corresponding plaintexts $(\bar{P}_j, \bar{P}'_j)$ (stored during Step 2.).
   For each of the 2^{10} candidates $k_0[5, 10, 15] \parallel \ell$, there is on average a single value for $k_0[0]$ that is compatible with the first pair $(\bar{P}_j, \bar{P}'_j)$. Finally, verify the 2^{10} candidates $k_0[0, 5, 10, 15] \parallel \ell$ with the remaining pairs $(\bar{P}_j, \bar{P}'_j)$. If a candidate k_0 is compatible with all the quartets, return it as the correct key.

Complexity. The attack queries new strucures of plaintext until a structure includes a right forward pair, and at least one quartet follows the backward characteristic. Each forward pair defines 2^{32} quartets, and the backward probability is 2^{-32}, therefore a right forward pair is detected with probability $1 - 1/e \approx 0.63$. When a right forward pair passes this condition, the attack succeeds for the correct guess of ℓ and m.

A structure of 2^{32} plaintexts contains $2^{32+31-6} = 2^{57}$ pairs with one colliding byte in the main diagonal of input. Therefore, we expect $2^{57-81.4} = 2^{-24.4}$ right pairs per structure, and we expect to iterate over $2^{24.4}/0.63 \approx 2^{25}$ structures on average before finding a right quartet. This corresponds to $2^{25+57-61.4} = 2^{20.6}$ candidates (P, P') at Step 2. and $2^{20.6+32-8} = 2^{44.6}$ candidates $(\bar{P}_i, \bar{P}'_i)$ at Step 3.

In Step 3, we consider on average $2^{20.6}$ pairs (P, P'), and each of the 2^{24} pairs $(\bar{P}_i, \bar{P}'_i)$ colliding in input byte a. First, we iterate over 2^{10} candidates $k_0[5, 10, 15]$, and verify that byte ℓ is inactive in w_0 for $(\bar{P}_i, \bar{P}'_i)$. This requires 6 S-box evaluations for each candidate. Then, we consider 2^{10} state differences in z_4 of $(P, \bar{P}_i)$ and deduce 2^{10} corresponding candidates for $k_6[0, 7, 10, 13]$. This

Algorithm 3 Low time complexity boomerang attack on 6-round AES.

```
loop                                    ▷ The expected number of iteration is $2^{25}$
    Query the encryption of a strucuture of $2^{32}$ plaintexts
    for all pairs $(P, P')$ colliding on one input byte, and two output anti-diagonals
do
        Denote the inactive byte position of $P + P'$ as $a$
        for $0 \le \ell < 4$ do
            Assume $w_0[\ell]$ is inactive for pairs $(P, P')$
            Deduce a set $S_\ell$ of $2^{16}$ candidates for $k_0[5, 10, 15]$
        for $0 \le i < 2^{32}$ do
            Define $\bar{C}_i = C + \nabla_i$, $\bar{C}'_i = C' + \nabla_i$,
            Query $\bar{P}_i = E^{-1}(\bar{C}_i), \bar{P}'_i = E^{-1}(\bar{C}'_i)$
        for all pairs $(\bar{P}_i, \bar{P}'_i)$ colliding on input byte $a$ do
            for $0 \le \ell < 4$ do
                Assume $w_0[\ell]$ is inactive for pairs $(P, P')$ and $(\bar{P}_i, \bar{P}'_i)$
                Filter the set $S_\ell$ to keep $2^8$ candidates for $k_0[5, 10, 15]$
            for $0 \le \tilde{m} < 4$ do
                Assume $z_4$ of pair $(P, \bar{P}_i)$ is active only on position $\tilde{m}$
                Deduce $2^8$ key candidates for $k_6[0, 7, 10, 13]$
            for all candidates for $\tilde{m} \parallel k_6[0, 7, 10, 13]$ do
                Gather 7 extra quartets s.t. $z_4$ of pair $(P, \bar{P}_i)$ is active only on position $m$
                for all candidates for $\ell \parallel k_0[5, 10, 15]$ do
                    Recover one candidate for $k_0[0]$ using one extra quartet
                    Use remaining quartets to verify $k_0[0, 5, 10, 15]$
```

requires 4 table lookups in the DDT for each candidate. In total, Step 3. requires on average $2^{20.6+24} \times (6 \times 2^{10} + 4 \times 2^{10}) = 2^{57.9}$ lookups.

In Step 4, fetching another pair $(\bar{P}_j, \bar{P}'_j)$ compatible with the characteristic requires 4 AES S-box calls and 1 lookup in the plaintext/ciphertext table of Step 2: compute the first column in z_4 of C from the first anti-diagonal of the ciphertext C (only once, this cost is amortized), shift by any value on byte of position $\tilde{m}$, and compute back the corresponding ciphertext $\bar{C}_j$. Since all 2^{32} ciphertexts $\bar{C}_i$ were queried during the decryption phase, $\bar{C}_j$ belongs to the table of queried ciphertexts. In total, this costs $2^{20.6+24} \times 4 \times 2^{10} = 2^{56.6}$ calls and $2^{20.6+24+10} = 2^{54.6}$ lookups in the plaintext/ciphertext table, for each of the 8 additional ciphertexts, i.e. $2^{59.6}$ S-box calls, and $2^{20.6+24+10+3} = 2^{57.6}$ lookups in the plaintext/ciphertext table.

The most costly part of the attack is the recovery of 1 average value of $k_0[0]$ from another pair $(\bar{P}_j, \bar{P}'_j)$, given $k_0[5, 10, 15] \parallel \ell$. To do so, we compute the difference in $y_0[5, 10, 15]$ for the pair $(\bar{P}_j, \bar{P}'_j)$ with 6 S-box evaluations, recover the only difference $y_0[0]$ yielding the correct difference pattern in w_0, and fetch the 1 average value $k_0[0]$ satisfying the transition $\bar{P}_j[0] + \bar{P}'_j[0] \to y_0[0]$. In total, this requires 7 lookups per pair. Then, given another pair $(\bar{P}_j, \bar{P}'_j)$, we verify that the difference in byte ℓ of w_0 is inactive with 8 S-box lookups for each of the 2^{10} candidates $k_0[0, 5, 10, 15] \parallel \ell$. We expect on average 2^2 remaining candidates for $k_0[0, 5, 10, 15] \parallel \ell$, and the rest of the filtering is of negligible complexity. In total, this requires $(7 + 8) \times 2^{20.6+24+10+10} = 2^{68.5}$ lookups, which corresponds to $2^{61.9}$ encryptions.

Further optimization. We observe that we process several pairs corresponding to the same candidate (P, P') and the same guess of $\tilde{m} \parallel k_6[0, 7, 10, 13]$. Indeed, for each (P, P'), we consider 2^{24} pairs $(\bar{P}_i, \bar{P}'_i)$, and 2^{10} values of $\tilde{m} \parallel k_6[0, 7, 10, 13]$ for each. On average, there is one pair $(\bar{P}_i, \bar{P}'_i)$ for each guess of $\tilde{m} \parallel k_6[0, 7, 10, 13]$, but we expect only 0.63×2^{34} distinct values. Pairs corresponding to repeated values of $\tilde{m} \parallel k_6[0, 7, 10, 13]$ are not necessary: assuming that (P, P') is a right forward pair, we only need one pair $(\bar{P}_i, \bar{P}'_i)$ corresponding to the correct guess of $\tilde{m} \parallel k_6[0, 7, 10, 13]$ for the attack to succeed. Therefore, we reduce the complexity by creating a table of size 2^{34} at the beginning of Step 3, and marking candidates $\tilde{m} \parallel k_6[0, 7, 10, 13]$ at the beginning of Step 4 to avoid processing the same guess twice. This reduces the complexity by a factor 0.63, leading to a total complexity of $0.63 \times 2^{61.9} = 2^{61.2}$. The storage of the table of 2^{34} bits corresponds to 2^{30} 128-bit states (if each bit is stored on one byte); it is not the memory bottleneck.

The complexity can potentially be further reduced by requiring the first two pairs $(\bar{P}_j, \bar{P}'_j)$ to be inactive on one byte of the main diagonal of $\bar{P}_j + \bar{P}'_j$. This increases the cost of locating the pairs by a factor 2^6, resulting in $2^{63.6}$ DDT lookups, and $2^{61.6}$ plaintext/ciphertext lookups in total to gather the first two pairs. If this is a negligible in comparison with the rest of attack[6], this

[6] This assumption actually depends on the architecture on which the attack is implemented: if the time required for a lookup in a table of size 2^{32} is large, this technique does not result in a better attack.

reduces the number of lookups required to test a key candidate from $7 + 8$ to $5 + 6$ because one S-box is inactive. This reduces the total complexity to $(5 + 6) \times 2^{20.6+24+10+10} \times 0.63 = 2^{67.5}$ lookups, which corresponds to $2^{60.9}$ encryptions.

Total complexity. The data complexity corresponds to 2^{57} encryptions and $2 \times 2^{20.6+32} = 2^{53.6}$ decryptions, for a total of $2^{57.1}$ queries. The memory complexity is bounded by the storage of the 2×2^{32} plaintexts $\bar{P}_i, \bar{P}'_i$, for $0 \leq i < 2^{32}$ from a forward pair (P, P').

This gives:

$$(D, T, M) = (2^{57.1}, 2^{60.9}, 2^{33})$$

Experimental verification. We implemented a simplified variant of the attack against a reduced AES with 4-bit S-boxes, and the experimental results match the theoretical analysis. The code is available on GitHub[7].

5 Incompatibility in a 6-Round Distinguisher

Several previous works presented *structural distinguishers* on 6-round AES with complexities between 2^{84} and 2^{97} (see [3] and the references therein). At first glance, it seems that the boomerang characteristics presented in the previous sections can easily be modified to obtain a distinguishing attack with a significantly lower complexity of about 2^{76}. 'All one has to do' is to modify the forward characteristic such that after the first round the difference is active only in a single byte, in order to improve the filtering in the backward direction. The resulting characteristic is presented in Figure 8.

However, a closer inspection shows that this 'distinguisher' is in fact flawed. The reason is that the characteristic is actually *impossible* due to an internal inconsistency in the forward trail between z_2 and w_2, as is shown in the figure.

We note that the same flaw appears in the *shifting retracing boomerang* attack on 5-round AES presented in [13, App. C3]. In that attack, one considers pairs of plaintexts for 5-round AES with non-zero difference only in bytes 0,5,10,15, and filters out all pairs except for those whose output difference in bytes 0,7,10,13 is equal to 0 or δ_L (for some fixed δ_L). Then, the remaining pairs are further analyzed, and it is claimed that the proportion of analyzed pairs is 2^{-31} of the number of initial pairs. However, in fact there is no point in analyzing pairs with zero ciphertext difference in bytes 0,7,10,13 since such pairs cannot have non-zero difference in a single byte after the first round (due to the same internal inconsistency as the one shown in Figure 8), and so they cannot be part of right boomerang quartets. As a result, the number of right quartets is reduced by a factor of 2, and to compensate for this, the data, time and memory complexities of the attack should be increased by a factor of $2^{0.5}$.

In order to help avoiding this failure in the future, we include this unsuccessful attempt here.

[7] https://github.com/Cryptosaurus/improved_boomerang

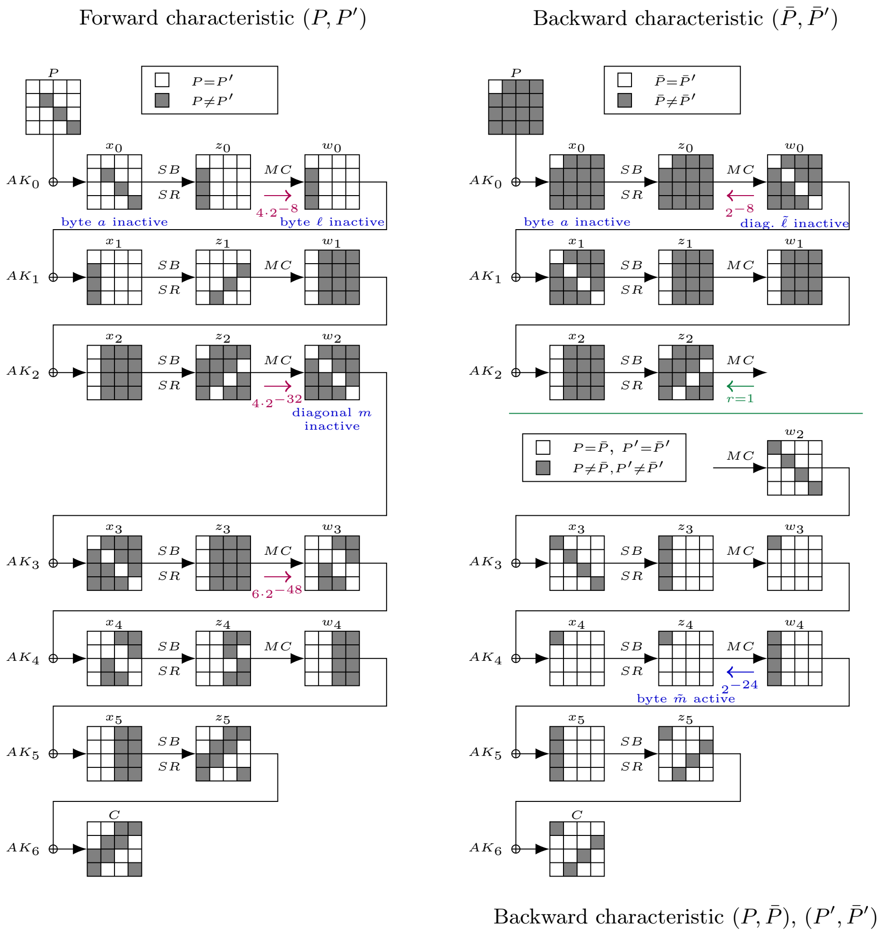

Fig. 7. Boomerang characteristic on 6-round AES with low time complexity.

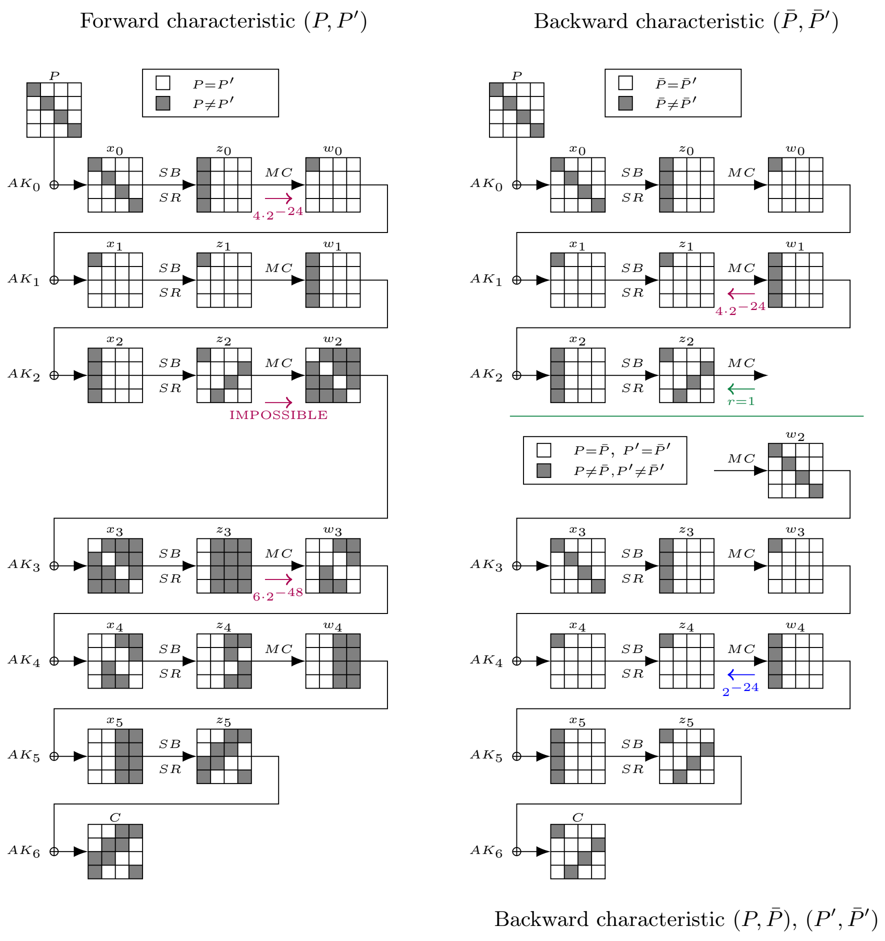

Fig. 8. An impossible forward pattern for a distinguishing attack.

6 Conclusions

In this paper we showed that elements of the retracing boomerang [14] and the truncated boomerang [3] techniques can be combined, to obtain the lower time complexity of the truncated boomerang attack along with the lower memory complexity of the retracing boomerang attack. The resulting attack we obtained is arguably the best boomerang attack on 6-round AES, and is the second best attack on 6-round AES after the Square attack. We hope that the new combined technique will be useful in boomerang attacks on other ciphers, and in particular, on ciphers which use 6-round AES as a component.

We conclude with a comment on relation to previous work. As was described in Section 3, the retracing boomerang framework of [14] contains two types of attacks: *shifting retracing boomerang*, in which most of the ciphertext pairs are discarded and the remaining pairs are used to generate new ciphertexts by a shifting process, and *mixing retracing boomerang*, in which all ciphertext pairs are used and the new ciphertexts are obtained by a mixing process (see Figure 4). In [14], all competitive attacks used the mixing retracing technique, while the shifting retracing technique had only 'proof-of-concept' examples. Our attacks here use the *shifting retracing* technique, and thus, provide the first competitive application of this technique. If we used the mixing retracing approach instead, our attack would become very similar to the *boomeyong* attack of Rahman *et al.* [25] on 6-round AES, whose complexity is about 2^{79}. In this case, the 'shifting retracing' approach yields significantly better results.

Acknowledgements

Part of the research was conducted in the framework of the workshop 'New directions in the cryptanalysis of AES', supported by the European Research Council under the ERC starting grant agreement n. 757731 (LightCrypt). The authors thank all the participants of the workshop for valuable discussions and suggestions. The first author is supported by the DGA. The second and fifth authors were supported in part by the Center for Cyber, Law, and Policy in conjunction with the Israel National Cyber Directorate in the Prime Minister's Office and by the Israeli Science Foundation through grants No. 880/18 and 3380/19. The third author was supported by the European Research Council under the ERC starting grant agreement n. 757731 (LightCrypt) and by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office. The fourth author is partially supported by ANR grants ANR-20-CE48-001 and ANR-22-PECY-0010.

References

1. Bao, Z., Guo, C., Guo, J., Song, L.: TNT: How to tweak a block cipher. In: Advances in Cryptology - EUROCRYPT 2020. Lecture Notes in Computer Science, vol. 12106, pp. 641–673. Springer (2020)

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: Advances in Cryptology - CRYPTO 2018. Lecture Notes in Computer Science, vol. 10992, pp. 185–212. Springer (2018)
3. Bariant, A., Leurent, G.: Truncated boomerang attacks and application to AES-based ciphers. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 3–35. Springer (2023)
4. Biham, E., Dunkelman, O., Keller, N.: The rectangle attack—rectangling the serpent. In: Advances in Cryptology—EUROCRYPT 2001: International Conference on the Theory and Application of Cryptographic Techniques Innsbruck, Austria, May 6–10, 2001 Proceedings 20. pp. 340–357. Springer (2001)
5. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. J. Cryptol. **4**(1), 3–72 (1991). `https://doi.org/10.1007/BF00630563`, `https://doi.org/10.1007/BF00630563`
6. Biryukov, A.: The boomerang attack on 5 and 6-round reduced AES. In: Dobbertin, H., Rijmen, V., Sowa, A. (eds.) Advanced Encryption Standard - AES, 4th International Conference, AES 2004, Bonn, Germany, May 10-12, 2004, Revised Selected and Invited Papers. Lecture Notes in Computer Science, vol. 3373, pp. 11–15. Springer (2004). `https://doi.org/10.1007/11506447_2`, `https://doi.org/10.1007/11506447_2`
7. Biryukov, A., Khovratovich, D.: Related-key cryptanalysis of the full AES-192 and AES-256. In: Advances in Cryptology - ASIACRYPT 2009. Lecture Notes in Computer Science, vol. 5912, pp. 1–18. Springer (2009)
8. Cho, J., Choi, K.Y., Dinur, I., Dunkelman, O., Keller, N., Moon, D., Veidberg, A.: WEM: A new family of white-box block ciphers based on the Even-Mansour construction. In: Topics in Cryptology - CT-RSA 2017. Lecture Notes in Computer Science, vol. 10159, pp. 293–308. Springer (2017)
9. Cid, C., Huang, T., Peyrin, T., Sasaki, Y., Song, L.: Boomerang connectivity table: a new cryptanalysis tool. In: 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 37. pp. 683–714. Springer (2018)
10. Daemen, J., Rijmen, V.: The design of Rijndael, vol. 2. Springer (2002)
11. Derbez, P., Euler, M., Fouque, P.A., Nguyen, P.H.: Revisiting related-key boomerang attacks on aes using computer-aided tool. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 68–88. Springer (2022)
12. Dunkelman, O., Ghosh, S., Keller, N., Leurent, G., Marmor, A., Mollimard, V.: Partial sums meet FFT: improved attack on 6-round AES. Eurocrypt 2024, to appear (2024)
13. Dunkelman, O., Keller, N., Ronen, E., Shamir, A.: The retracing boomerang attack. Journal of Cryptology, to appear. Available at Cryptology ePrint Archive (2019)
14. Dunkelman, O., Keller, N., Ronen, E., Shamir, A.: The retracing boomerang attack. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 280–309. Springer (2020)
15. Dunkelman, O., Keller, N., Shamir, A.: A practical-time related-key attack on the kasumi cryptosystem used in gsm and 3g telephony. In: Advances in Cryptology–CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings 30. pp. 393–410. Springer (2010)
16. Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D.A., Whiting, D.: Improved cryptanalysis of Rijndael. In: Fast Software Encryption, FSE 2000. Lecture Notes in Computer Science, vol. 1978, pp. 213–230. Springer (2000)

17. Grassi, L.: Probabilistic mixture differential cryptanalysis on round-reduced AES. In: Selected Areas in Cryptography - SAC 2019. Lecture Notes in Computer Science, vol. 11959, pp. 53–84. Springer (2019)
18. Jean, J.: TikZ for Cryptographers. `https://www.iacr.org/authors/tikz/` (2016)
19. Jean, J., Nikolic, I., Peyrin, T.: Tweaks and keys for block ciphers: The TWEAKEY framework. In: Advances in Cryptology - ASIACRYPT 2014. Lecture Notes in Computer Science, vol. 8874, pp. 274–288. Springer (2014)
20. Jean, J., Nikolic, I., Peyrin, T., Seurin, Y.: The deoxys AEAD family. J. Cryptol. **34**(3), 31 (2021)
21. Keliher, L., Sui, J.: Exact maximum expected differential and linear probability for two-round Advanced Encryption Standard. IET Inf. Secur. **1**(2), 53–57 (2007)
22. Kelsey, J., Kohno, T., Schneier, B.: Amplified boomerang attacks against reduced-round mars and serpent. In: International Workshop on Fast Software Encryption. pp. 75–93. Springer (2000)
23. Mennink, B., Neves, S.: Optimal PRFs from blockcipher designs. IACR Trans. Symmetric Cryptol. **2017**(3), 228–252 (2017)
24. Murphy, S.: The return of the cryptographic boomerang. IEEE Transactions on Information Theory **57**(4), 2517–2521 (2011)
25. Rahman, M., Saha, D., Paul, G.: Boomeyong: Embedding yoyo within boomerang and its applications to key recovery attacks on AES and Pholkos. IACR Trans. Symmetric Cryptol. **2021**(3), 137–169 (2021)
26. Rønjom, S., Bardeh, N.G., Helleseth, T.: Yoyo tricks with AES. In: ASIACRYPT 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10624, pp. 217–243. Springer (2017)
27. of Standards, U.N.I., Technology: Advanced Encryption Standard. Federal Information Processing Standards publication no. 197 (2001)
28. Wagner, D.: The boomerang attack. In: International Workshop on Fast Software Encryption. pp. 156–170. Springer (1999)
29. Yan, X., Tan, L., Xu, H., Qi, W.: Improved mixture differential attacks on 6-round AES-like ciphers towards time and data complexities. Journal of Information Security and Applications **80**, 103661 (2024)
30. Zhang, W., Wu, W., Feng, D.: New results on impossible differential cryptanalysis of reduced AES. In: Information Security and Cryptology - ICISC 2007. Lecture Notes in Computer Science, vol. 4817, pp. 239–250. Springer (2007)