Key Structures: Improved Related-Key Boomerang Attack against the Full AES-256

Jian Guo[1], Ling Song[2], and Haoyang Wang[3(✉)]

[1] Nanyang Technological University, Singapore
guojian@ntu.edu.sg
[2] Jinan University, China
songling.qs@gmail.com
[3] Shanghai Jiao Tong University, China
haoyang.wang@sjtu.edu.cn

Abstract. This paper introduces structure to key, in the related-key attack settings. While the idea of structure has been long used in key-recovery attacks against block ciphers to enjoy the birthday effect, the same had not been applied to key materials due to the fact that key structure results in uncontrolled differences in key and hence affects the validity or probabilities of the differential trails. We apply this simple idea to improve the related-key boomerang attack against AES-256 by Biryukov and Khovratovich in 2009. Surprisingly, it turns out to be effective, *i.e.*, both data and time complexities are reduced by a factor of about 2^8, to 2^{92} and 2^{91} respectively, at the cost of the amount of required keys increased from 4 to 2^{19}. There exist some tradeoffs between the data/time complexity and the number of keys. To the best of our knowledge, this is the first essential improvement of the attack against the full AES-256 since 2009. It will be interesting to see if the structure technique can be applied to other AES-like block ciphers, and to tweaks rather than keys of tweakable block ciphers so the amount of required keys of the attack will not be affected.

Keywords: AES, differential, boomerang, key structure, related key

1 Introduction

The Birth of AES. After the *Data Encryption Standard* (DES) was attacked by differential cryptanalysis due to Biham and Shamir [7,8] and later by linear cryptanalysis due to Matsui [29,30], the U.S. National Institute of Standards and Technology (NIST) initiated the public AES competition (1997 – 2000), out of which Rijndael [16] designed by Daemen and Rijmen won the competition and became officially the *Advanced Encryption Standard* in 2001. There are three variants, *i.e.*, AES-k with $k \in \{128, 192, 256\}$ denoting the key sizes in bits. AES became *de facto* the most popular and important block cipher in the world now for data protection, widely adopted by both industry and government agencies.

The computation power nowadays is still far from breaking AES by bruteforce, even against the smallest variant AES-128, and due to the existence of AES-256, it will remain sound even under attack by the future quantum computers. Hence, a much longer lifespan is expected if no security flaw is discovered.

The Security. Since the design of Rijndael, AES has attracted tremendous efforts from the research community in security analysis. One of the most important security features of AES is its *proven* resistance against differential and linear cryptanalysis, which were applied to its predecessor — DES. It achieved this by the so-called *wide trail strategy* [15], *e.g.*, the minimum number of active S-boxes (those with non-zero differences) in 4 consecutive AES rounds for any differential characteristic in the single-key setting is 25. It has been analyzed by many cryptanalysis techniques[4], just to name a few here. Biryukov, Khovratovich, and Nikolić gave the first key-recovery attack against AES-256 by differentials [11] in 2009, the complexity of which was later improved in [10] by using related-key boomerang attack. In [28], Lu *et al.* gave 7-round attacks against AES-128 and AES-192, and 8-round against AES-256 by using impossible differentials. Leveraging integral cryptanalysis, Ferguson *et al.* [20] gave a practical attack against 6-round AES and then the first attack against 7-round AES. In terms of complexities of the key-recovery attacks in the single-key setting, the best attacks up to date are due to the Demirci-Selçuk meet-in-the-middle attack [17, 18, 19]. The meet-in-the-middle attack, which was previously known to be powerful for finding preimages of hash functions, led to attacks against 7-round AES in some hashing modes [31]. Many more attacks were found under other attack settings *e.g.*, single-key, related-key, hashing modes, for various security aspects, *e.g.*, key-recovery, collision/preimage finding in hashing modes, distinguisher, etc. Up to date, the most successful key-recovery[5] attack against AES-128/192/256 is for 7, 9, and 9 out of 10, 12, and 14 rounds respectively in the single-key setting, due to Derbez *et al.* [18] and an improvement by Li *et al.* [27].

The Boomerang Attack. To the best of our knowledge, the best attack against AES, in terms of number of attacked rounds, is due to Biryukov *et al.* [10, 11] which dates back to 2009, where the key of the full version of AES-256 and AES-192 can be recovered under the related (sub-)key setting. While the bound given by the wide trail strategy could not be overcome in the differential attack under the single-key setting, differential characteristics with much higher probability exist in the related-key settings, where the differences from round keys and the data path can be cancelled out. Also, the boomerang attack is able to utilize two high-probability differential characteristics for small number of rounds. The attack succeeds due to these two properties.

Our Contributions. Biryukov *et al.*'s works remain as the best publicly known key-recovery attack against the full version of AES-256, and there exists no

[4] Only a few papers are cited here as examples since there are simply too many results.
[5] Besides those optimized brute-force style attacks, such as [12].

essential improvement since 2009. In this paper, we try to improve their attack, in terms of data and time complexities, under the same related-subkey boomerang attack framework. The core idea comes from the observation that, while *structure* has been used in plaintext to enjoy the birthday effect and improve cryptanalysis, the same could potentially be applied to key material as well, even though this has not been tried yet. It is necessary to note that similar expressions to "key structure" already existed in some papers before, such as [6]. Their purpose is to generate some required subkey differences from key structure, due to non-linear key schedule of their targeted ciphers. However, our aim in this paper is to make further improvement by enjoying birthday effect of a key structure. There are many technical difficulties to overcome in the key structure boomerang attack framework, before the idea can eventually work out.

- Firstly, when *structure* rather than a fixed or chosen difference is introduced in the key material, one has to ensure that the uncontrollable difference in the key will not affect the validity of the two differential characteristics in the data path of the boomerang attack.
- Secondly, when *structure* is applied to the key, one has to ensure that the two differentials are neutral to each other in two halves of the key schedule, i.e., the differential characteristics in the key schedule in one half will not affect the other half regardless of the actual key difference chosen from the structure.
- Thirdly, one has to ensure that the key difference will not affect the probability of the differential characteristics. Note that Biryukov *et al.* chose the high probability difference transition of the S-box (those with 2^{-6}, rather than the 2^{-7} ones for the AES S-box) in order to increase the overall probability of the differential characteristics. In our case of structure, we only utilize those 2^{-6} difference pairs to achieve the same optimization. Although this only happens once in every 2^8 difference pairs (so we lose most of the pairs), we gain back by enforcing the high probability 2^{-6} transition once, and re-use it multiple times in the differential characteristics.

The final result, presented in Section 4, turns out to improve both the time and data complexities of Biyukov *et al.* attack by a factor of about 2^8, at the expense of the number of required keys being increased from 4 to 2^{19}. A detailed comparison is provided in Table 1.

Organization. The rest of the paper is organized as follows. Section 2 gives the necessary preliminaries for understanding the attack, and Section 3 explains the ideas behind key structures. The details of the improved attack are given in Section 4. Finally, Section 5 concludes the paper.

2 Preliminaries

2.1 Description of AES

The Advanced Encryption Standard(AES) [16] is an iterated block cipher which encrypts 128-bit plaintext with secret key of sizes 128, 192, and 256 bits. Its

Table 1. Comparison with previous key-recovery attacks on full AES-256

Attack	Time	Data	Memory	# keys	Reference
Related-Key Differential	2^{131}	2^{131}	2^{65}	2^{35}	[11]
Related-Key Boomerang	$2^{99.5}$	$2^{99.5}$	2^{77}	4	[10]
Key-Structure	$2^{92.5+s}$	2^{92+s}	2^{89-s}	2^{17-s}	Ours
Boomerang	2^{92+s}	2^{91+s}	2^{89-s}	2^{19-s}	

Note: $0 \leq s \leq 7.5$

internal state can be represented as a 4×4 matrix whose elements are byte value (8 bits) in a finite field of $GF(2^8)$. The round function consists of four basic transformations in the following order:

- SubBytes (SB) is a nonlinear substitution that applies the same S-box to each byte of the internal state.
- ShiftRows (SR) is a cyclic rotation of the i-th row by i bytes to the left, for $i = 0, 1, 2, 3$.
- MixColumns (MC) is a multiplication of each column with a Maximum Distance Separable (MDS) matrix over $GF(2^8)$.
- AddRoundKey (AK) is an exclusive-or with the round key.

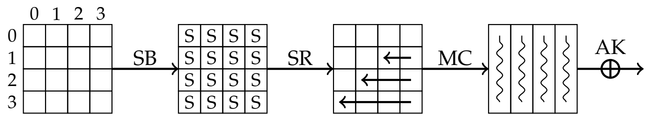


Figure 1. AES round function

At the very beginning of the encryption, an additional whitening key addition is performed, and the last round does not contain MixColumns. AES-128, AES-192, and AES-256 share the same round function with different number of rounds: 10, 12, and 14, respectively.

The key schedule of AES transforms the master key into subkeys which are used in each of the rounds. Here, we describe the key schedule of AES-256. The 256-bit master key is divided into 8 32-bit words $(W[0], W[1], ..., W[7])$, then $W[i]$ for $i \geqslant 8$ is computed as

$$W[i] = \begin{cases} W[i-8] \oplus \mathtt{SB}(\mathtt{RotByte}(W[i-1])) \oplus Rcon[i/8] & i \equiv 0 \bmod 8, \\ W[i-8] \oplus \mathtt{SB}(W[i-1]) & i \equiv 4 \bmod 8, \\ W[i-8] \oplus W[i-1] & \text{otherwise} \end{cases}$$

The i-th *subkey* is of size 256-bit denoted by K^i, K^0 is the master key. `RotByte` is a cyclic shift by one byte to the left, and *Rcon* is the round constant. The key schedule of AES-128 and AES-192 is slightly different due to the different key sizes, since this paper does not focus on these two variants, we refer to [16] for details.

Property of the AES S-box. The details of the S-box and the Difference Distribution Table (DDT) could be found in [16]. For any input difference $\Delta_{in} \neq 0$, there exists exactly one Δ_{out} such that $\texttt{DDT}(\Delta_{in}, \Delta_{out}) = 4$ (this results in the highest probability 2^{-6} for the AES S-box transition), 126 values of Δ_{out} such that $\texttt{DDT}(\Delta_{in}, \Delta_{out}) = 2$ (*i.e.*, probability 2^{-7}), and the rest 129 values of Δ_{out} with $\texttt{DDT}(\Delta_{in}, \Delta_{out}) = 0$. Those $(\Delta_{in}, \Delta_{out})$'s with $\texttt{DDT}(\Delta_{in}, \Delta_{out}) \neq 0$ are called *compatible*, and others are *incompatible*. These statistics will be used in our attack later.

2.2 Boomerang Attack

The boomerang attack was introduced in [33]. It regards the target cipher as a composition of two sub-ciphers E_0 and E_1. The first sub-cipher is supposed to have a differential $\alpha \rightarrow \beta$, and the second one to have a differential $\gamma \rightarrow \delta$, with probabilities p and q, respectively. The basic boomerang attack requires an adaptive chosen plaintext/ciphertext scenario, and plaintext pairs result in a right quartet with probability p^2q^2. It works with four plaintext/ciphertext pairs $(P_1, C_1), (P_2, C_2), (P_3, C_3), (P_4, C_4)$, and the basic attack procedure is as follows. The attacker queries the encryption oracle with the input P_1 and $P_2 = P_1 \oplus \alpha$ to obtain C_1 and C_2, and calculate $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$, which are sent to the decryption oracle to obtain P_3 and P_4. Later, Kelsey *et al.* [24] developed the amplified boomerang which is pure chosen-plaintext attack and a right quartet is obtained with probability $p^2q^22^{-n}$. Further, it was pointed out in [4, 5, 33] that any value of β and γ is allowed as long as $\beta \neq \gamma$. As a result, the probability of the right quartet is increased to $2^{-n}\hat{p}^2\hat{q}^2$, where $\hat{p} = \sqrt{\Sigma_i \Pr^2(\alpha \rightarrow \beta_i)}$ and $\hat{q} = \sqrt{\Sigma_j \Pr^2(\gamma_j \rightarrow \delta)}$. This improved attack framework is named the *rectangle attack*.

Related-Key Boomerang Attack. Boomerang and rectangle attacks under related-key setting were formulated in [6, 25, 26]. Let ΔK and ∇K be the key differences for E_0 and E_1, respectively. The attacker needs to access four related-key oracles with $K_1 \in \mathbb{K}$, where $\mathbb{K}$ is the key space, $K_2 = K_1 \oplus \Delta K$, $K_3 = K_1 \oplus \nabla K$ and $K_4 = K_1 \oplus \Delta K \oplus \nabla K$. In the related-key boomerang attack, paired plaintexts P_1, P_2 such that $P_1 \oplus P_2 = \alpha$ are queried to K_1 encryption oracle and K_2 encryption oracle, and the attacker receives ciphertexts C_1 and C_2. Then C_3 and C_4 are calculated by $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$, and then queried to K_3 decryption oracle and K_4 decryption oracle. The resulting plaintext difference $P_3 \oplus P_4$ equals to α with probability p^2q^2. Related-key rectangle attacks can be similarly formulated.

Boomerang Switch and Boomerang Connectivity Table. The boomerang switch was used to gain free rounds in the middle of the cipher in the attacks against the full AES-192 and AES-256 [10]. The idea was to optimize the transition between the differential characteristics of E_0 and E_1 in order to minimize the overall complexity of the distinguisher. In [10], three types of switch were introduced which are the *Feistel switch*, the *ladder switch* and the *S-box switch*. These switches were further generalized in the *boomerang connectivity table* (BCT) [14].

In this paper, we utilize the ladder switch to optimize our attack. The idea of the ladder switch is to realize that a cipher can be decomposed into smaller parallel transformations instead of rounds by default. The principle can be explained in the framework of BCT, see Figure 2 with the case when $\Delta \neq 0$ and $\nabla = 0$. For any values of x_1 and x_2, with difference Δ, their outputs after S-box application are y_1 and y_2, respectively. Since the boomerang shift happens when $\nabla = 0$, we have $y_3 = y_1 \oplus \nabla = y_1$ and $y_4 = y_2 \oplus \nabla = y_2$. Thus, after the inversed S-box is applied, the paired values (x_3, x_4) is equal to (x_1, x_2) with probability 1, *i.e.*, the returned pair will always have difference Δ. The same also holds when $\Delta = 0$ and $\nabla \neq 0$.

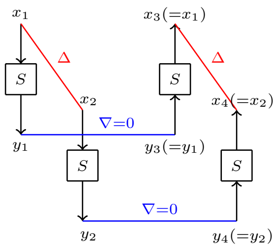


Figure 2. The ladder switch on a single S-box

2.3 Notations

The byte at i-th row, j-th column of an internal state a is denoted by $a_{i,j}$, as illustrated in Figure 1, where i and j start from 0. We refer to the byte of plaintext by $p_{i,j}$, the byte of the r-th subkey by $k^r_{i,j}$, and the byte of the r-th internal state after `SubBytes` by $x^r_{i,j}$. For the differential characteristics of boomerang distinguisher, we denote the difference used in E_0 by Δ and the difference in E_1 by ∇.

3 Key Structures

In differential cryptanalysis as [8], the attacker tries to find a distinguisher of a cipher so that he can distinguish the cipher from a random permutation. Then, key recovery attacks can be mounted based on the distinguisher directly, or with additional rounds added before and/or after the distinguisher. In this paper, we focus on the latter one.

We assume that there is a distinguisher which consists of a differential $\alpha \rightarrow \beta$ with probability p covering the last r_1 rounds of the target cipher. In order to launch a full-round attack (r rounds), $(r - r_1)$ rounds should be prepended to the distinguisher. The aim of the attacker is to obtain enough ciphertext pairs with difference β by querying the encryption oracle with pre-chosen plaintexts. We define V to be the space spanned by all the plaintext differences that may lead to the difference α after the first $(r - r_1)$ rounds, and let $m = log_2|V|$.

Structure from Plaintext. The first step of the attack is to generate pairs of plaintexts whose output differences after the first $(r-r_1)$ rounds are the expected input difference α of the differential. The way to improve the efficiency of this step is to build a structure of plaintexts which consists of $P \oplus v_i$, where P is chosen randomly and $v_i \in V$. The XOR difference between any two elements of the structure belongs to V. In this way, at most 2^{2m-1} unordered plaintext pairs (P_i, P_j) (the order of i, j does not matter) can be composed from a single structure, while the data and time complexity to prepare this structure is only 2^m. We refer to the ratio between the number of pairs generated and the size of structure as *gain* (this is also called birthday effect in other places), quantitatively it is $2^{(2m-1)-m} = 2^{m-1}$ here. If more plaintext pairs are needed for the attack, another new structure can be constructed in the same way by selecting another random value of P. However, this would not increase the overall gain. As can be seen, the gain only depends on the structure size. Thus, if the structure size could be increased, the attack complexity would be reduced accordingly.

Structure from Key. In the related-key setting [3], the attacker is allowed to choose a desired relation between keys. The most common form is: for an unknown key K_1, the attacker uses a XOR difference D to produce another key $K_2 = K_1 \oplus D$. Then, the subkey additions in round functions can be used to cancel some differences in the differential attack in order to obtain better differential characteristics. Compared with the single-key setting, the related-key setting provides additional freedom in choosing the key difference D. In our case, this fact enlightens us that a key structure utilizing the key difference could be used to improve the attack.

For a related-key differential characteristic with key difference D, we can build a key structure from the original secret key K_1, and let $\{K_1 \oplus v_i \mid v_i \in V_D\}$ be the set of keys inside the structure, where V_D is the space consisting of the differences that have the same truncated difference as that of D. Similarly, we define $m_k = log_2|V_D|$. Together with a plaintext structure, at most a total of $2^{2(m+m_k)-1}$ unordered pairs of $((P_i, K_i), (P_j, K_j))$ can be obtained, while only 2^{m+m_k} data/queries are used. Thus, the gain increases to $2^{2(m+m_k)-1-(m+m_k)} = 2^{m+m_k-1}$, compared to the use of plaintext structure alone.

The Use of Key Structure. Key structure should be applied together with plaintext structure to provide additional advantage. However, compared to plaintext structure, key structure has more constraints in its application. Due to the fact that each plaintext in the plaintext structure will be encrypted with each key in the key structure during an attack, the difference between the pairs of keys will not be fixed, thus the subkey differences in the whole rounds are difficult to control and are unlikely to match the exact differential characteristic $\alpha \to \beta$.

Hence, in order to have as many valid key pairs as possible, the key schedule is better to be linear or the proportion of the non-linear part is small, and the key difference should not have strong impact on the truncated differential characteristic, which means that the truncated differential characteristic should be able to be instantiated with many differences. Otherwise, if the differential characteristic is valid with only a small proportion of key pairs, smaller than 2^{-m_k}, the use of key structure will only weaken the attack. On the other hand, the distinguisher obtained with key structure might not be as good as the original one without key structure, because the probability of the differential characteristic will vary according to key difference since the propagation of non-linear part in the internal state will be different. All in all, in order to make good use of key structure, the extra data and time consumption of it should be lower than the gain it offers.

Last but not least, it is necessary to mention that at most one key structure can be constructed in an attack as the key structure is created from the original secret key which is fixed.

4 Improved Boomerang Attack on AES-256

In this section, we apply key structure to the related-key boomerang attack on AES-256, which is based on the attack in [10]. We will first give an overview of the boomerang distinguisher, then describe the construction of the key structure, and finally explain the details of the attack.

The differential characteristics used in our boomerang attack are depicted in Figure 3. The differential characteristic of E_1 is fixed, while the differential characteristic of E_0 has a lot of candidates. In Figure 3, different colors refer to different values. The differentials for all the active S-boxes of the differential characteristic of E_1 are set to be $(\texttt{0x01}, \texttt{0x1f})$, which holds with probability 2^{-6}. For the differential characteristic of E_0, the red and blue hashed cells are not fixed but always pass the S-box differential with the maximal probability 2^{-6}, and the green cells are unknown. The switching position of the boomerang distinguisher is pointed by the green ovals.

The differences in the key schedule are given in Table 2. Since the differential characteristic of E_0 is not fixed, we use the variables R, B and G_i to represent its truncated pattern, $i = 1, 2, 3, 4$. Given the value of R, B is then derived from the table `DDT` with the requirement that $\texttt{DDT}(R, B) = 4$, lastly the value of G_i is uniquely determined by B through the `MixColumns` transformation:

$$\begin{pmatrix} 0 \\ B \\ 0 \\ 0 \end{pmatrix} \xRightarrow{\texttt{MixColumns}} \begin{pmatrix} G_1 \\ G_2 \\ G_3 \\ G_4 \end{pmatrix}.$$

Table 2. Key schedule difference for the boomerang attack on AES-256. The values are given in hexadecimal notation.

ΔK^i					
0	? 00 00 00 G_1 G_1 G_1 G_1	1	00 00 00 00 G_1 00 G_1 00	2	00 00 00 00 G_1 G_1 00 00
	? R R R ? G_2 G_2 G_2		00 R 00 R G_2 00 G_2 00		00 R R 00 G_2 G_2 00 00
	? 00 00 00 G_3 G_3 G_3 G_3		00 00 00 00 G_3 00 G_3 00		00 00 00 00 G_3 G_3 00 00
	? 00 00 00 G_4 G_4 G_4 G_4		00 00 00 00 G_4 00 G_4 00		00 00 00 00 G_4 G_4 00 00
3	00 00 00 00 G_1 00 00 00	4	00 00 00 00 G_1 G_1 G_1 G_1		
	00 R 00 00 G_2 00 00 00		00 R R R ? ? ? ?		
	00 00 00 00 G_3 00 00 00		00 00 00 00 G_3 G_3 G_3 G_3		
	00 00 00 00 G_4 00 00 00		00 00 00 00 G_4 G_4 G_4 G_4		

∇K^i					
0	? ? ? ? ? 00 ? 00	1	? 01 ? 00 ? ? 00 00	2	? ? 00 00 ? 00 00 00
	X X X X 1f 00 1f 00		X 00 X 00 1f 1f 00 00		X X 00 00 1f 00 00 00
	? ? ? ? 1f 00 1f 00		? 00 ? 00 1f 1f 00 00		? ? 00 00 1f 00 00 00
	? ? ? ? 21 00 21 00		? 00 ? 00 21 21 00 00		? ? 00 00 21 00 00 00
3	? 01 01 01 3e 3e 3e 3e	4	01 00 01 00 3e 00 3e 00	5	01 01 00 00 3e 3e 00 00
	X 00 00 00 1f 1f 1f 1f		00 00 00 00 1f 00 1f 00		00 00 00 00 1f 1f 00 00
	? 00 00 00 1f 1f 1f 1f		00 00 00 00 1f 00 1f 00		00 00 00 00 1f 1f 00 00
	? 00 00 00 21 21 21 21		00 00 00 00 21 00 21 00		00 00 00 00 21 21 00 00
6	01 00 00 00 3e 00 00 00	7	01 01 01 01 ? ? ? ?		
	00 00 00 00 1f 00 00 00		00 00 00 00 1f 1f 1f 1f		
	00 00 00 00 1f 00 00 00		00 00 00 00 1f 1f 1f 1f		
	00 00 00 00 21 00 00 00		00 00 00 00 21 21 21 21		

4.1 Construction of the Key Structure

The key relation used in our attack is a complex form that allows the attacker to choose a desired XOR difference of a subkey at any round. This setting is also defined as the related-subkey setting in [9].

Now, we describe how to construct the key structure, denoted by S_k. The key structure is generated from the second subkey K^1. One can observe in Figure 3 that the difference of the second 256-bit subkey has 10 active bytes, but the difference cannot be chosen randomly. The differential characteristic of E_0, as well as the one of E_1, is constructed following the idea of local collision [13], that is, once a subkey difference is added to the internal state, the next subkey difference will try to cancel it in the next round. Therefore, in order to generate such a differential characteristic for E_0, the two active bytes in the first half of

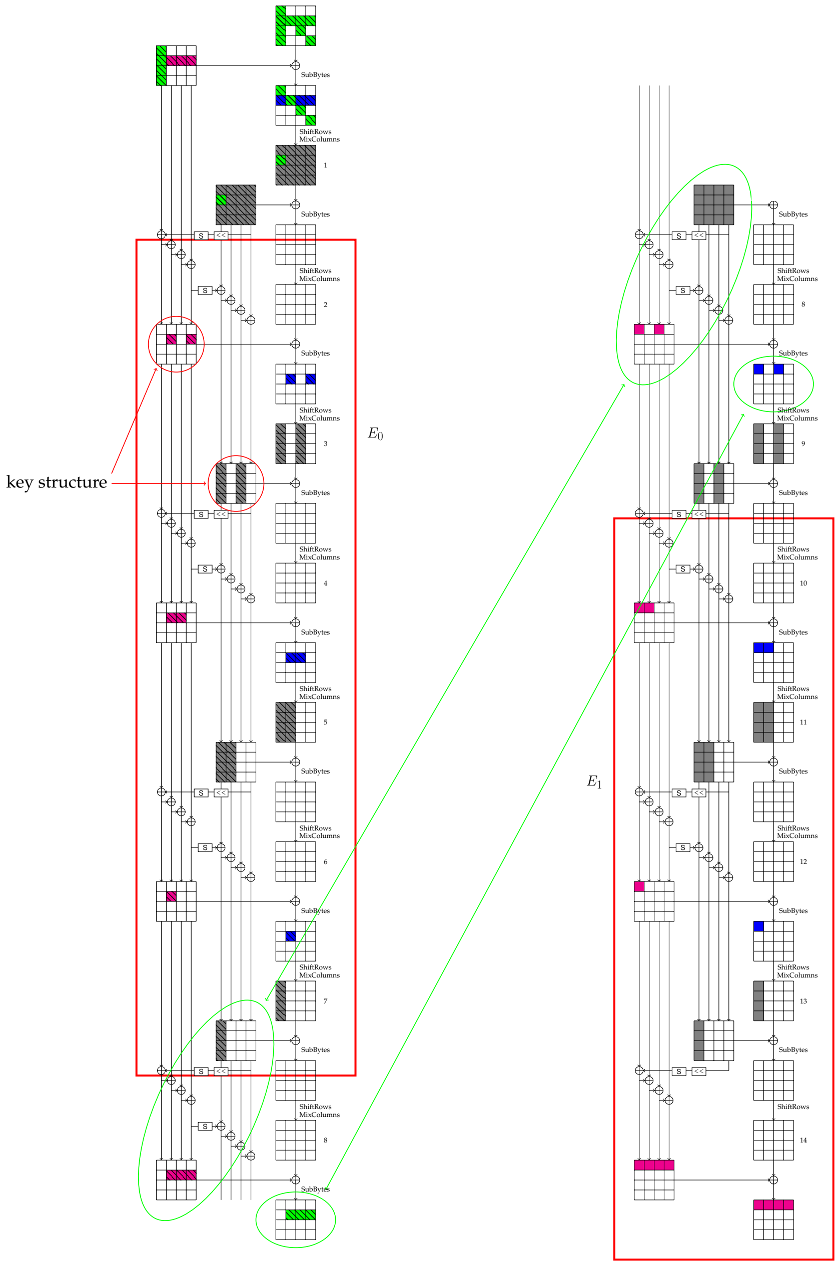


Figure 3. The differential characteristics of the boomerang attack against AES-256

ΔK^1 must take the same difference value, thus it can choose 2^8 values at most. Besides, for the second half of ΔK^1, the differences of the two active columns are also required to be the same value, because the two active columns are supposed to cancel the two active columns in the internal state according to the differential characteristic of E_0, and each active column in the internal state is computed from a single active byte through `MixColumns` and the two active bytes are equal. Furthermore, this relation also implies that the two active columns of ΔK^1 can only choose 2^8 values at most. To sum up, there are 2^{16} valid values for ΔK^1, each of which is denoted by ΔK_i^1, $1 \le i \le 2^{16}$.

For a secret key K_0, the key structure S_k is generated by adding the non-zero difference ΔK_i^1 to the second subkey of K_0, from which a new secret key K_i can be uniquely determined, see Figure 4(a). Finally, the key structure consists of 2^{16} keys, from which 2^{31} unordered key pairs can be composed.

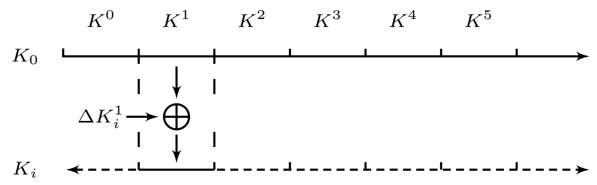
Note that the keys in the key structure are used in the encryption side of the boomerang attack. For the key K_i' used in the decryption side, they are computed by adding the fixed difference ∇K to the jointed state of the second half of K_i^3 and the first half of K_i^4, then the full K_i' can be uniquely determined by the obtained eight consecutive columns, see Figure 4(b). By doing so, the differential characteristic of E_1 will be fixed. The actual value of ∇K can be found in Table 2, it will make sure that the differential characteristic of E_1 is the optimal one *i.e.*, all S-box transitions happen with probability 2^{-6}.

Given a pair of keys (K_A, K_B) chosen from the key structure S_k, together with the corresponding key pair (K_A', K_B') used in the decryption side, the four keys form a key quartet. For a key quartet, the differences in the key schedule for both the differential characteristics of E_0 and E_1 can be found in Table 2. In particular, for the differential characteristic of E_1 (where the key pair (K_A, K_A') or (K_B, K_B') is applied), some byte in ∇K^i for $i = 1, 2, 3$ can even be determined due to the slow diffusion of the key schedule. These values will play an important role in the following key recovery attack. Last but not least, we note that only one key structure is used in our attack.

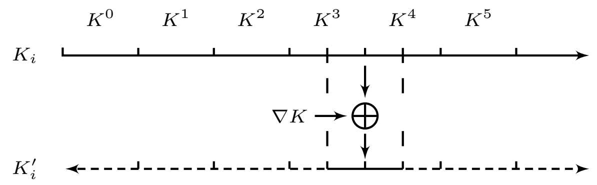
4.2 Boomerang Distinguisher

Let us compute the probability of the boomerang distinguisher covering rounds 2–14. For the differential characteristic of E_0 which covers rounds 2–8, there are 5 active S-boxes and the differentials $(\Delta_{in}, \Delta_{out})$ for all of them are the same. Because of the use of key structure, the values for both Δ_{in} and Δ_{out} are not fixed, but they are directly related to the subkey differences, which are determined by the key pair used in the differential characteristic. Among the total of 2^{31} key pairs that can be composed from the key structure, $(2^7 - 2)/2^8$ of it will make the differential $(\Delta_{in}, \Delta_{out})$ happen with probability 2^{-7}, while a proportion of $1/2^8$ will lead the probability to 2^{-6}. Accordingly, the 5 active S-boxes in rounds 2–8 are passed with probability $2^{-7\times 5} = 2^{-35}$ for the first case and $2^{-6\times 5} = 2^{-30}$ for the second case.

The boomerang is switched in round 9. Although the differential characteristic of E_0 is not fixed, its truncated pattern is uniquely determined. Accordingly,



(a) Generating key structure S_k



(b) Computing K_i' from K_i

Figure 4. Key generation

we can ensure that there is no overlapped active S-box in round 9 between the differential characteristics of E_0 and E_1. Thus, according to the BCT, the two differential characteristics are compatible for the boomerang attack and the switching probability is 1. Besides, it was reported recently that the boomerang switch can actually happen in multiple rounds in [32,34], so we have also verified the switching effect in rounds 8–10, and it matches our evaluation.

For the differential characteristic of E_1, there are 3 active S-boxes in rounds 10–14. Note that only one differential characteristic is used for E_1 and the differentials for all the active S-boxes are optimal with probability 2^{-6}, thus the probability of the differential characteristic of E_1 of rounds 10–14 is $2^{-6\times 3} = 2^{-18}$.

Finally, the probability of the boomerang distinguisher is either $2^{2\times(-35-18)} = 2^{-106}$ or $2^{2\times(-30-18)} = 2^{-96}$, depending on the key pair from the key structure.

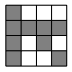
4.3 A Detailed Description of the Attack

One round is added at the beginning of the boomerang distinguisher to launch the full-round attack. The plaintext difference pattern, as show in Figure 3, is deduced from both the first subkey difference and the internal state difference in the second round. The attack procedure is described in Algorithm 1.

For each key pair, we can compose 2^{144} plaintext pairs from 1 plaintext structure, out of which $2^{144-72} = 2^{72}$ will pass through the first round with the desired input difference of the boomerang distinguisher. In total, 2^{103} pairs pass the first round for all the 2^{31} key pairs. The probability of the boomerang distinguisher is 2^{-106} for a proportion of $(2^7-2)/2^8$ key pairs, thus around $2^{103}\cdot(2^7-2)/2^8\cdot 2^{-106} \approx 2^{-4}$ right quartets are expected. On the other hand, $2^{103}\cdot 1/2^8 \cdot 2^{-96} = 2^{-1}$ right quartets are expected when the probability of the boomerang distinguisher is 2^{-96} for $1/2^8$ of key pairs. Compared to the first case,

Algorithm 1: Related-key boomerang attack on AES-256 using key structure

```
Prepare a plaintext structure consisting of $2^{72}$ plaintexts, which traverses all
  values of the 9 gray cells in [grid], and takes arbitrary constants in the others.
Create a hash table $H$ of size $2^{88}$.
for each of the $2^{16}$ keys $K$ in the key structure $S_K$ do
    for each of the $2^{72}$ plaintexts $P$ in the plaintext structure do
        Encrypt $P$ under key $K$, denote the ciphertext as $C$.
        Compute $C' = C \oplus \Delta C$.
        Decrypt $C'$ with $K'$, $K'$ being computed from the corresponding $K$,
          and denote the new plaintext by $P'$.
        Insert the plaintext pair $(P, P')$ into the hash table $H$, indexed by the 7
          bytes of $P'$ where constants of the plaintext structure fall and 2 bytes
          of $P \oplus P'$ at positions $(2, 0)$ and $(3, 0)$.
    end
end
```



the boomerang distinguisher in the second case is much better, so we will only adopt the second one in our attack. Therefore, in order to obtain 4 right quartets, 2^3 plaintext structures are required. We need to repeat Algorithm 1 2^3 times with different plaintext structures and the same key structure.

In the following, we will explain how to gradually filter out wrong quartets and recover key bits. Let us compute the number of quartet candidates after Algorithm 1. Firstly, there is a 56-bit filter at the output of the boomerang. Then, observe in Table 2 that $\nabla k^0_{i,7} = 0$ for $i > 1$, so $\Delta k^0_{i,0}$ should be equal for both pairs (K_A, K_B) and (K'_A, K'_B) in a key quartet, which implies that $\Delta p_{i,0}$ should be equal for both plaintext pairs (P_1, P_2) and (P'_1, P'_2) for the right quartet as well, because $\Delta k^0_{i,0}$ is equal to $\Delta p_{i,0}$ according to the differential characteristic of E_0. This is a 16-bit filter. So there are on average $2^{72+16-56-16} = 2^{16}$ collisions for each index of the hash table H, from which 2^{31} quartets can be composed. In total, $2^{31+72} = 2^{103}$ quartet candidates are left for all the 2^{72} indices of the hash table, and thus 2^{106} quartet candidates for all the 2^3 plaintext structures. These candidates are further filtered by the following steps. The key bytes that can be recovered are listed in Figure 5.

Step I. Note that the key pairs of a right quartet must meet the requirement that the active S-boxes in the differential characteristic of E_0 are passed with probability 2^{-6}. The requirement is satisfied with probability 2^{-8}, thus $2^{106-8} = 2^{98}$ quartet candidates are eligible. Now we explain how to obtain the 2^{98} quartet candidates. The differential of the active S-boxes can be deduced by the difference in the key schedule. For example, $\Delta k^1_{1,1}$ is the input difference of the S-box at

7							5
3	4	2	2	4D			7
D		4					5
D			4				5

Figure 5. The AES-256 key state with key information obtained at each step. Digits stand for the sub-steps in step II, "D" means difference.

the position $(1,1)$ in the third round, and $\Delta k^1_{2,4}$ is the corresponding output difference due to the `ShiftRows` and `MixColumns`. Thus, we can simply check whether the differential $(\Delta k^1_{1,1}, \Delta k^1_{2,4})$ is optimal for the AES S-box. Once it is confirmed, the differentials of the 5 active S-boxes will also be determined. Note that the differential characteristics of E_0 used in both sides of the boomerang are the same, thus we only need to check the encryption side, the details are given in Algorithm 2.

Algorithm 2: Filtration in Step I

```
for each index of the hash table $H$ do
    Insert the $2^{16}$ collisions into a new hash table $H'$ indexed by the difference
      of $k^1_{1,1}$ between the current key and the original key $K_0$.
    for each index $i$ of $H'$ do
        Insert the $2^{16-8} = 2^8$ collisions (on average) into a new hash table $H''_i$
          indexed by the difference of $k^1_{2,4}$ between the current key and the
          original key $K_0$.
    end
    for index $i$ of $H'$ from 0 to $2^8 - 2$ do
        for index $j$ of $H'$ from $i$+1 to $2^8 - 1$ do
            Compute $\Delta k^1_{1,1} = i \oplus j$, and find the value of $\Delta k^1_{2,4}$ such that
              $\mathtt{DDT}(\Delta k^1_{1,1}, \Delta k^1_{2,4}) = 4$.
            for each index $s$ of $H''_i$ do
                Compute $t = s \oplus \Delta k^1_{2,4}$. Check whether $t$ is in $H''_j$. If yes, the
                  pairs of $H''_i$ and $H''_j$ compose quartet candidates.
            end
        end
    end
end
```

Step II. There are $2^{31-8} = 2^{23}$ key quartets remaining after Step I, and each has on average $2^{98-23} = 2^{75}$ quartet candidates. In the following steps, we will proceed with each key quartet independently, and use K_A, K_B, K'_A and K'_B to denote the four keys.

1) There is a 2-bit filter at $\Delta p_{1,2}$ and $\Delta p_{1,3}$ due to the S-box compatibility, thus 4-bit at both sides of the boomerang in total. Besides, there is also a 2-bit filter at $\Delta p_{2,0}$ and $\Delta p_{3,0}$ due the the S-box compatibility in the key schedule. Thus, the number of quartets is reduced to $2^{75-6} = 2^{69}$.
2) Each quartet proposes 2^2 candidates of $k^0_{1,2}$ and $k^0_{1,3}$ for K_A and K'_A each, thus there are in total 2^4 candidates. As can be seen from Table 2 , the four "X" of ∇K^0 are equal and take only 2^7 values. Hence, the differences $\Delta k^0_{1,2}$ and $\Delta k^0_{1,3}$ between K_A and K'_A have to be equal to X, which is a 16-bit filter for the key candidates. Thus, $2^{69+4+7-16} = 2^{64}$ quartet candidates are left, and the values of $k^0_{1,2}$ and $k^0_{1,3}$ are suggested.
3) Observe that the value of $\Delta k^0_{1,0}$ is determined by $k^0_{2,7}$ and $\nabla k^0_{2,7} = 0$ from Table 2, the values of $\Delta k^0_{1,0}$ should be the same for both the key pairs (K_A, K_B) and (K'_A, K'_B). Since $\Delta k^0_{2,7}$ is known, $\Delta k^0_{1,0}$ can take 2^7 values. For each guess of $\Delta k^0_{1,0}$, it has to be compatible with $\Delta p_{1,0}$ and $\Delta x^0_{1,0}$ through the S-box, which is a 2-bit filter for both sides of the boomerang. After that, each quartet proposes two candidates of $k^0_{1,0}$ for K_A and K'_A, respectively. Moreover, there is an 8-bit filter because the difference $\nabla k^0_{1,0}$ between K_A and K'_A should be equal to X. In the end, the number of quartet candidates is reduced to $2^{64+7-2+2-8} = 2^{63}$, and the value of $k^0_{1,0}$ is suggested.
4) Notice that $\nabla k^1_{1,3} = 0$, a reasoning similar to the one above can be applied to $\Delta k^0_{1,4}$, which can take 2^7 values. For each guess of $\Delta k^0_{1,4}$, the values of $\Delta x^0_{0,0}, \Delta x^0_{1,1}, \Delta x^0_{2,2}, \Delta x^0_{3,3}$ can be uniquely computed by inverting the `MixColumns` transformation. There is a 1-bit filter on $\Delta x^0_{1,1}, \Delta x^0_{2,2}, \Delta x^0_{3,3}$ each due to the S-box compatibility, then 6-bit filter in total on both sides of the boomerang. Each quartet proposes two candidates of $k^0_{1,1}, k^0_{2,2}, k^0_{3,3}$ for K_A and K'_A, respectively. However, the difference $\nabla k^0_{1,1}$ between K_A and K'_A is restricted to X, which results in an 8-bit filter. To summarize, the number of quartets is reduced to $2^{63+7-6+6-8} = 2^{62}$ and the values of $k^0_{1,1}$, $k^0_{2,2}$, $k^0_{3,3}$ as well as $\Delta k^0_{1,4}$ are suggested.
5) Since $\Delta k^0_{1,0}$, $\Delta k^0_{2,0}$ and $\Delta k^0_{3,0}$ are known, it will provide 2 guesses for each of $k^0_{2,7}$, $k^0_{3,7}$ and $k^0_{0,7}$. However, these guesses for K_A and K'_A are the same because the differences $\nabla k^0_{2,7}$, $\nabla k^0_{3,7}$ and $\nabla k^0_{0,7}$ between K_A and K'_A are all 0. Thus, in this step, the number of key candidates is increased to $2^{62+3} = 2^{65}$, and the values of $k^0_{2,7}$, $k^0_{3,7}$ and $k^0_{0,7}$ are suggested.
6) Note that the key bytes $k^0_{2,7}$, $k^0_{1,0}$, $k^0_{1,1}$, $k^0_{1,2}$, $k^0_{1,3}$ and the difference $\Delta k^0_{1,4}$ have been derived in the above steps. On the other hand, we notice that $\Delta k^0_{1,4}$ can be computed from $k^0_{2,7}$, $k^0_{1,0}$, $k^0_{1,1}$, $k^0_{1,2}$ and $k^0_{1,3}$ according to the key schedule. This constraint can provide an 8-bit filter, and thus the number of key proposals is reduced to 2^{57}.
7) Make a guess of $k^0_{1,7}$ of K_A, which has 2^8 choices, then $k^0_{1,7}$ will be known for all the four keys and $\Delta k^0_{0,0}$ can be computed. After that, for each side of the boomerang there is a 1-bit filter on $\Delta k^0_{0,0}$ due to the S-box compatibility, then each quartet will propose 2 candidates of $k^0_{0,0}$ for K_A and K'_A, respectively. Thus $2^{57+8-2+2} = 2^{65}$ key proposals are obtained.

In the end, 2^{65} key candidates are proposed and 11 key bytes for each of K_A, K_B, K'_A and K'_B are suggested. However, many bytes are strongly related according to Table 2. Among them, at least $k^0_{0,0}$, $k^0_{1,1}$, $k^0_{2,2}$ and $k^0_{3,3}$ of K_A and K'_A are independent, so we can recover 15 bytes with 2^{65} proposals for each key quartet, and thus 2^{88} proposals for all the 2^{23} key quartets. Additionally, differences of 3 bytes are recovered: $\Delta k^0_{1,4}, \Delta k^0_{2,0}, \Delta k^0_{3,0}$, where $\Delta k^0_{2,0}$ and $\Delta k^0_{3,0}$ can be directly obtained from plaintext difference, but they can not be used to derive the corresponding key bytes.

Recover the Key. Recall that 4 right quartets are expected for the attack, which are supposed to be distinguishable from other wrong quartets. However, the 4 right quartets are very likely to be combined with different key pairs due to the key structure, thus the correct key bytes proposed by them will belong to different keys, which is hard to be distinguished. So we have to deduce all the proposed key bytes to the original keys K_0 and K'_0. Looking at Table 2, we can see that most bytes of the first subkey difference between K_0 and all the other keys are known, except $\Delta k^0_{1,4}$ and $\Delta k^0_{i,0}$ where $0 \leq i \leq 3$. Hence, for those key bytes whose differences are known, all the proposals can be deduced to K_0. The same reasoning applies to K'_0. As for the two unknown differences $\Delta k^0_{0,0}$ and $\Delta k^0_{1,0}$, since the value of $k^0_{1,7}$ was derived and the difference of $\Delta k^0_{1,7}$ is known, the difference $\Delta k^0_{0,0}$ can be easily computed through the key schedule. Same trick holds for $\Delta k^0_{1,0}$, it can be computed from $k^0_{2,7}$ and $\Delta k^0_{2,7}$. Then, we can deduce the proposals of the two bytes to K_0 and K'_0.

In the end, all the proposed key bytes can be deduced to K_0 and K'_0. We have 2^{88} proposals for 120 key bits, and the correct proposal is supposed to appear 4 times. The probability that a wrong key is suggested 4 times is $\binom{2^{88}}{4} \cdot (2^{-120})^4 \cdot (1 - 2^{-120})^{2^{88}-4} \approx 2^{-138.5}$, thus the expect number of such a key is $2^{120-138.5} = 2^{-18.5}$, while the 4 right quartets would always vote for the correct one. Therefore, no wrong key will survive and the correct 120 key bits will be recovered. With the knowledge of the recovered key bits, the remaining part of the key can be found with many approaches, which will not dominate the cost of the whole attack.

Complexity. In our attack, a total of 2^{92} plaintexts and ciphertexts are generated, thus the data complexity is 2^{92}. In Algorithm 1, there are 2^{88} encryption oracle calls, 2^{88} XOR operations, and 2^{88} decryption oracle calls. As the plaintexts are added into the hash table, each plaintext requires one memory access, thus 2^{88} memory accesses in total. Thus, for the 2^3 plaintext structures, the total time complexity of Algorithm 1 is 2^{92} encryption/decryption oracle calls and 2^{91} memory accesses.

In Algorithm 2, the plaintext pairs (P, P') in H are added into many new hash tables, which requires $2^{72} \times (2^{16} + 2^{16}) = 2^{89}$ memory accesses. The lookups in `DDT` require $2^{72+15} = 2^{87}$ memory accesses. The lookups in the hash table H''_i require $2^{72+23} = 2^{95}$ memory accesses, which dominates the algorithm. In sum, for the entire 2^3 plaintext structures, Step I requires $2^{95+3} = 2^{98}$ memory accesses.

For Step II 1), it requires 1 memory access for each quartet to check whether these bytes are compatible. Therefore, the time complexity of this step is 2^{98} memory accesses.

After Step II 1), the number of remaining quartets is 2^{92}, and the number is continuously decreasing in the following steps, thus the following computation will not dominate the cost of the whole attack.

Following the idea from [5,9,20], where memory access can be converted to equivalent amount of encryption/decryption oracle calls, one AES-256 encryption/decryption is equivalent to roughly 2^8 memory accesses by counting the number of S-box lookups. Therefore, the 2^{98} memory accesses to the hash table H_i'' in Step I can be converted to 2^{90} encryptions/decryptions, and the same can be done for the 2^{98} memory accesses in Step II 1). Finally, we conclude that the whole attack requires 2^{92} plaintexts and ciphertexts, the time complexity being equivalent to $2^{92.5}$ encryptions (2^{92} encryptions/decryptions and 2^{99} memory accesses $\approx 2^{91}$ encryptions), and the memory complexity is 2^{89}.

Further Improvement. In the attack, the key structure is only added in K_A and K_B in E_0, the same idea could be extended to the K_A' and K_B' in E_1. However, this attempt could not work. For a right quartet, the differentials of E_0 in both sides of the boomerang should be the same, which determines that the differential characteristic in the key schedule of (K_A', K_B') must be the same as that of (K_A, K_B). In order to meet this condition, the switch in the key schedule should be the same for a quartet, *i.e.*, the differences ∇K added to K_A and K_B at the middle of the fourth and fifth subkey should be the same. Even if we use a structure of ∇K, we still need to find right quartets for each value of ∇K separately. Thus, this method does not provide additional improvement to the attack.

Although the potential key structure is a failure, it can also be used to improve the attack slightly. As discussed above, the functionality of the key structure is equivalent to plaintext structure: using a different ∇K leads to a different set of (P_1', P_2'). In our attack, 2^3 plaintext structures are required to produce 4 right quartets. Instead of choosing 2^3 plaintext structures, we could also choose 2^3 ∇K (There are $2^8 - 1$ values of ∇K to produce the optimal differential characteristic of E_1). In this way, 2^{88} encryptions and $2^{88+3} = 2^{91}$ decryptions are needed, thus the number of decryptions dominates the complexity. The time and data complexity is reduced to 2^{92} and 2^{91}, respectively. The number of keys will increase to $2^{16+3} = 2^{19}$.

More Tradeoffs. One key structure consists of 2^{16}, and hence the total number of keys required in this attack is 2^{17} (2^{16} for the encryption and decryption oracles each). There is a tradeoff between the number of keys required, and the time and data complexities, by reducing the size of the key structure, *i.e.*, with a 2^{16-s} key structures, the resulted complexities of the attack will be: Time 2^{92+s}, Data 2^{91+s}, and # Keys 2^{17-s}, for $0 \leq s \leq 16$. Further to note, when $s \geq 7.5$

the time complexity becomes higher than that in [10], and our attack offers no more advantage, so the tradeoff makes sense only for $0 \leq s \leq 7.5$.

5 Conclusion

In this paper, we brought the idea of structures to key materials, and successfully applied it to the related-key boomerang attack against AES-256. This improved the best known attack against AES-256 by reducing the data/time complexities by a factor of about 2^8, at the cost of more required keys. While the general principle is simple, its deployment contains many details and it is important to ensure that the introduction of key structure will not invalidate or significantly reduce the probability of the differential characteristics. More tradeoffs are provided between time/data complexity and the number of required keys.

Other Potential Applications. We note that our structure technique was applied to key material, and hence increases the number of required keys for the attack to succeed. However, this may be avoided when the attack is applied to AES-based tweakable block ciphers so that the structure is applied to tweak, rather than keys. There are two such cases: TAES [1] and the TBCs following TWEAKEY framework [22]. TAES is basically AES-256 with the concatenation of a 128-bit secret key and a 128-bit tweak as the 256-bit key input. The TWEAKEY framework treats the key and tweak in the same way and names the combined input "tweakey". Following it, there are several dedicated AES-like proposals such as the Deoxys-BC in the Deoxys AE design [23], SKINNY [2], and Kiasu [21]. Users will have the choice to decide which are the bits to be used as key or tweak material. The potential application of our technique is that, when the structure is applied to the tweak of either TAES or AES-based TWEAKEY designs, the increased requirement applies to the tweak only, and that of keys remains un-affected.

Inapplicability to AES-192 [10] and Differential Attack [11]. The boomerang attack was applied to AES-192 as well in [10], so the idea of key structure naturally applies. However, looking into the details, the key bytes recovered in the AES-192 attack falls in two different locations, in both the pre- and post-whitening keys. Note that in our improved attack on AES-256, we are able to deduce the count of key suggestions of all keys to the original key K_0 and K'_0, however this becomes impossible for both pre- and post-whitening keys simultaneously in case of AES-192. The direct application of the idea of key structure to the differential attack in [11] seems difficult, as the probability of the differentials will drop significantly, which overrules the potential gain key structures might brings. It will be interesting to see if these technical difficulties could be overcome and find more applications of key structures.

Acknowledgements. This research is partially supported by the Nanyang Technological University in Singapore under Grant 04INS000397C230, Singapore's

Ministry of Education under Grants RG91/20 and MOE2019-T2-1-060, the National Natural Science Foundation of China (Grants 62022036, 62132008, 62172410, 61732021), and the National Key Research and Development Program of China (Grant 2018YFA0704704).

References

1. Bao, Z., Guo, J., Iwata, T., Minematsu, K.: ZOCB and ZOTR: Tweakable Blockcipher Modes for Authenticated Encryption with Full Absorption. IACR Transactions on Symmetric Cryptology 2019(2), 1–54 (2019)
2. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS. In: Advances in Cryptology – CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 123–153 (2016)
3. Biham, E.: New Types of Cryptanalytic Attacks Using Related Keys. Journal of Cryptology 7(4), 229–246 (1994)
4. Biham, E., Dunkelman, O., Keller, N.: The Rectangle Attack - Rectangling the Serpent. In: Advances in Cryptology – EUROCRYPT 2001. LNCS, vol. 2045, pp. 340–357 (2001)
5. Biham, E., Dunkelman, O., Keller, N.: New Results on Boomerang and Rectangle Attacks. In: Fast Software Encryption – FSE 2002. LNCS, vol. 2365, pp. 1–16 (2002)
6. Biham, E., Dunkelman, O., Keller, N.: Related-Key Boomerang and Rectangle Attacks. In: Advances in Cryptology – EUROCRYPT 2005. LNCS, vol. 3494, pp. 507–525 (2005)
7. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. In: Advances in Cryptology – CRYPTO'90. LNCS, vol. 537, pp. 2–21 (1991)
8. Biham, E., Shamir, A.: Differential Cryptanalysis of the Full 16-Round DES. In: Advances in Cryptology – CRYPTO'92. LNCS, vol. 740, pp. 487–496 (1993)
9. Biryukov, A., Dunkelman, O., Keller, N., Khovratovich, D., Shamir, A.: Key Recovery Attacks of Practical Complexity on AES-256 Variants with up to 10 Rounds. In: Advances in Cryptology – EUROCRYPT 2010. LNCS, vol. 6110, pp. 299–319 (2010)
10. Biryukov, A., Khovratovich, D.: Related-Key Cryptanalysis of the Full AES-192 and AES-256. In: Advances in Cryptology – ASIACRYPT 2009. LNCS, vol. 5912, pp. 1–18 (2009)
11. Biryukov, A., Khovratovich, D., Nikolic, I.: Distinguisher and Related-Key Attack on the Full AES-256. In: Advances in Cryptology – CRYPTO 2009. LNCS, vol. 5677, pp. 231–249 (2009)
12. Bogdanov, A., Khovratovich, D., Rechberger, C.: Biclique Cryptanalysis of the Full AES. In: Advances in Cryptology – ASIACRYPT 2011. LNCS, vol. 7073, pp. 344–371 (2011)
13. Chabaud, F., Joux, A.: Differential Collisions in SHA-0. In: Advances in Cryptology – CRYPTO'98. LNCS, vol. 1462, pp. 56–71 (1998)
14. Cid, C., Huang, T., Peyrin, T., Sasaki, Y., Song, L.: Boomerang Connectivity Table: A New Cryptanalysis Tool. In: Advances in Cryptology – EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 683–714 (2018)
15. Daemen, J.: Cipher and Hash Function Design Strategies Based on Linear and Differential Cryptanalysis. Ph.D. thesis, Doctoral Dissertation, March 1995, KU Leuven (1995)

16. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography, Springer (2002)
17. Demirci, H., Selçuk, A.A.: A Meet-in-the-Middle Attack on 8-Round AES. In: Fast Software Encryption – FSE 2008. LNCS, vol. 5086, pp. 116–126 (2008)
18. Derbez, P., Fouque, P.A., Jean, J.: Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In: Advances in Cryptology – EUROCRYPT 2013. LNCS, vol. 7881, pp. 371–387 (2013)
19. Dunkelman, O., Keller, N., Shamir, A.: Improved Single-Key Attacks on 8-Round AES-192 and AES-256. In: Advances in Cryptology – ASIACRYPT 2010. LNCS, vol. 6477, pp. 158–176 (2010)
20. Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D., Whiting, D.: Improved Cryptanalysis of Rijndael. In: Schneier, B. (ed.) Fast Software Encryption – FSE 2000. LNCS, vol. 1978, pp. 213–230 (2001)
21. Jean, J., Nikolić, I., Peyrin, T.: KIASU v1. Additional first-round candidates of CAESAR compeition (2014)
22. Jean, J., Nikolic, I., Peyrin, T.: Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In: Advances in Cryptology – ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 274–288 (2014)
23. Jean, J., Nikolić, I., Peyrin, T., Seurin, Y.: Deoxys-II. Finalist of CAESAR compeition (2014)
24. Kelsey, J., Kohno, T., Schneier, B.: Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In: Fast Software Encryption – FSE 2000. LNCS, vol. 1978, pp. 75–93 (2001)
25. Kim, J., Hong, S., Preneel, B., Biham, E., Dunkelman, O., Keller, N.: Related-Key Boomerang and Rectangle Attacks: Theory and Experimental Analysis. IEEE transactions on information theory 58(7), 4948–4966 (2012)
26. Kim, J., Kim, G., Hong, S., Lee, S., Hong, D.: The Related-Key Rectangle Attack - Application to SHACAL-1. In: ACISP 04: 9th Australasian Conference on Information Security and Privacy. LNCS, vol. 3108, pp. 123–136 (2004)
27. Li, L., Jia, K., Wang, X.: Improved Single-Key Attacks on 9-Round AES-192/256. In: Fast Software Encryption – FSE 2014. LNCS, vol. 8540, pp. 127–146 (2015)
28. Lu, J., Dunkelman, O., Keller, N., Kim, J.: New Impossible Differential Attacks on AES. In: Progress in Cryptology - INDOCRYPT 2008: 9th International Conference in Cryptology in India. LNCS, vol. 5365, pp. 279–293 (2008)
29. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: Advances in Cryptology – EUROCRYPT'93. LNCS, vol. 765, pp. 386–397 (1994)
30. Matsui, M.: The First Experimental Cryptanalysis of the Data Encryption Standard. In: Advances in Cryptology – CRYPTO'94. LNCS, vol. 839, pp. 1–11 (1994)
31. Sasaki, Y.: Meet-in-the-Middle Preimage Attacks on AES Hashing Modes and an Application to Whirlpool. In: Fast Software Encryption – FSE 2011. LNCS, vol. 6733, pp. 378–396 (2011)
32. Song, L., Qin, X., Hu, L.: Boomerang Connectivity Table Revisited. Application to SKINNY and AES. IACR Transactions on Symmetric Cryptology 2019(1), 118–141 (2019)
33. Wagner, D.: The Boomerang Attack. In: Knudsen, L.R. (ed.) Fast Software Encryption – FSE'99. LNCS, vol. 1636, pp. 156–170 (1999)
34. Wang, H., Peyrin, T.: Boomerang Switch in Multiple Rounds. Application to AES Variants and Deoxys. IACR Transactions on Symmetric Cryptology 2019(1), 142–169 (2019)