Biclique Cryptanalysis of Full Round AES with Reduced Data Complexity

Donghoon Chang, Mohona Ghosh, Somitra Sanadhya

Indraprastha Institute of Information Technology
donghoon, mohonag,somitra@iiitd.ac.in

Abstract. Biclique cryptanalysis was proposed by Bogdanov et al. in Asiacrypt 2011 as a new tool for cryptanalysis of block ciphers. A major hurdle in carrying out biclique cryptanalysis is that it has a very high query complexity (of the order of 2^{88} for AES-128, 2^{80} for AES-192 and 2^{40} for AES-256). This naturally puts a big question mark over the practical feasibility of implementing biclique attack in real world. In this work, we re-evaluate the security of full round AES against biclique cryptanalysis. We describe an alternate biclique construction with significantly reduced query complexity (of the order of 2^{24} for AES-128, 2^{32} for AES-192 and 2^8 for AES-256) at the expense of a slightly increased computational cost. In our approach, we use independent biclique technique to launch a chosen ciphertext attack against AES.
Keywords: AES, block ciphers, cryptanalysis, biclique, meet-in-the middle, key recovery.

1 Introduction

AES (Advanced Encryption Standard), standardized by the US NIST in October 2000, has been accepted and adopted worldwide thereafter. It remains the most favored cryptographic scheme in both software and hardware applications. Despite the design having been subjected to tremendous scrutiny in the past 12 years, it has remained remarkably immune to all cryptanalytic attacks of practical significance.

Single key recovery attacks such as multiset attack [1], square attack [1], boomerang attack [2, 3], impossible differentials [4, 5], algebraic attack [6, 7] etc. could break only a limited number of rounds of AES (for all versions of AES). Related key attack models [8–10] delivered better results comparatively as they could be applied to breach the security guarantee of full rounds of AES-192 and AES-256. However, related key models require multiple pairs of related keys (keys which have some kind of common relation between them) which is practically a very strong requirement and hence is not considered a big threat to AES in real world [11].

Until recently, there was no single key model attack known which could break full AES-128 better than brute force attack. In Asiacrypt, 2011 Bogdanov et al. [11] proposed a novel idea called biclique attack which allows an attacker to recover AES secret key up to 3-5 times faster than exhaustive search. Biclique cryptanalysis had earlier been introduced by Khovratovich et al. in [12] for pre-image attack on hash functions Skein and SHA-2. The approach is a variant of the meet-in-the attack. The concept was taken over by Bogdanov et al. to AES cryptanalysis and has been subsequently adopted to break many other block ciphers such as ARIA [13], SQUARE [14], TWINE [15], HIGHT [16], PRESENT [17] etc. Though Bogdanov et al. successfully demonstrated that biclique attack reduced key recovery effort, it suffered from high data complexity (more so for AES-128 and 192) rendering it ineffective for practical implementations.

The ultimate value of any cryptanalysis technique lies in its successful real-world deployment. Biryukov et al. [10] stress upon the importance of practical feasibility of various cryptanalytic models while designing various attacks. They discuss several factors such as number of ciphertexts/plaintexts required (the lesser the better), known plaintext attack vs. chosen plaintext attack (former being more favored), single key model vs. related key model (single key model preferred) etc. which should be taken into consideration while developing the attacks. Though relative importance of a specific factor changes from scenario to scenario, all of them impact the total running time and data complexity of

an attack. Many a times theory suggests that an attack is possible but it never gets used in practice because the attack demands resources which are not within reachable bound limits. Unfortunately, biclique attack too falls under the bracket of such practically restricted techniques. Conventionally in an exhaustive key search, searching 2^k keys (where k is the size of the key in bits) for the actual key can be achieved with a single plaintext-ciphertext pair. If multiple key candidates satisfy the given pair then we need to make just one more encryption query (in the worst case few pairs of plaintext-ciphertext would suffice) and check the plaintext-ciphertext pairs to find the right key amongst the possible candidates. Overall the query complexity is very small (not more than 4-5 queries). On the other hand, biclique technique requires very high number of oracle queries (of the order of 2^{88} for AES-128 and 2^{80} for AES 192) making it unreachably huge even for an attacker having unlimited computational power. Bogdanov et.al in [11] state that, "We notice that the data complexity of key recovery can be significantly reduced by sacrificing only a small factor of computational advantage". However, no details or further explanation is provided on how and by what quantity can this reduction be obtained. This and the high data complexity in [11] motivated us to construct an alternate biclique and differential trail which brings the data complexity of this attack technique against AES to practical limits.

Our contribution: We use the independent biclique approach proposed in [11]. We slide the biclique construction of [11] down by one round in our attack. We observe that doing so not only brings a change in the differential trail and the key trail, but also produces a significant reduction in the total number of ciphertexts required for decryption oracle at the cost of a diminutive increase in the total running time.

- We construct a 2-round biclique for AES-128. Using this biclique, our attack against full round AES-128 has a total time complexity of $2^{126.5}$ and requires $\mathbf{2^{24}}$ ciphertexts.
- We construct a 3-round biclique for AES-192. Using this biclique, our attack against full round AES-192 has a total time complexity of $2^{190.5}$ and requires $\mathbf{2^{32}}$ ciphertexts.
- We construct a 3-round biclique for AES-256. Using this biclique, our attack against full round AES-256 has a total time complexity of $2^{254.72}$ and requires $\mathbf{2^{8}}$ ciphertexts.

We also find a couple of discrepancies in [11] in evaluating the cost of the attacks. The time complexity for attacking AES-192 and AES-256 has been given as $2^{189.74}$ and $2^{254.42}$ whereas our calculations peg these at $2^{190.13}$ and $2^{254.6}$ respectively.

Table 1 summarizes our attacks on AES and compares the complexity with the previous biclique attack on AES.

The paper is organized as follows. Section 2 gives a brief description of AES followed by Section 3 which explains the biclique key recovery attack. In Section 4 we present a biclique attack for the full round AES-128. Section 5 and 6 cover the biclique attack on full round AES-192 and AES-256 respectively. In Section 7 we summarize and conclude our paper.

2 Description of AES

AES is a block cipher which adopts the classical substitution-permutation network structure. The AES specification defines 3 key sizes - 128 bit, 192 bit and 256 bit with block size limited to a fixed 128 bit size for all the three alternatives. By design, AES is byte-oriented and follows operations in GF(2^8). Each AES variant has different number of rounds per full encryption, i.e. 10, 12 and 14 rounds for AES-128, AES-192 and AES-256 respectively. Each round consists of 4 steps: SubBytes, ShiftRows, MixColumns and AddRoundKey. AES operates on a state array of 4×4 byte matrix and key array of 4×4, 4×6 and 4×8 byte size respectively. For further information on AES, please refer to [1].

Table 1. Summary of our results and comparison with [11].

Algorithm	Rounds	Data Complexity	Computational Complexity	Biclique rounds	Reference
AES-128	10	2^{88}	$2^{126.1}$	3	[11]
		$\mathbf{2^{24}}$	$2^{126.5}$	2	This work, § 4
AES-192	12	2^{80}	$2^{189.74\dagger}$	4	[11]
		$\mathbf{2^{32}}$	$2^{190.5}$	3	This work, § 5
AES-256	14	2^{40}	$2^{254.42\ddagger}$	4	[11]
		$\mathbf{2^{8}}$	$2^{254.72}$	3	This work, § 6

† Our analysis at the end of § 5 estimates the cost as $2^{190.13}$.
‡ Our analysis at the end of § 6 estimates the cost as $2^{254.6}$.

To avoid confusion and facilitate comparison, we follow the same notation as adopted in [11]. Briefly, in a differential path # 1, #2 represent the state before SubBytes and after MixColumns for Round 1, #3, #4 represent the state before SubBytes and after MixColumns for Round 2 and so on. The 128 bit subkeys are denoted as \$0, \$1, \$2 .... and so on. Bytes are addressed column-wise (0-3#first column), (4-7#second column), (8-11#third column) and (12-15#fourth column). The i^{th} byte in state S is represented as S_i. The i^{th} byte in subkey \$ K is represented as $\$K_i$.

3 Biclique Cryptanalysis

Biclique attack is a kind of divide-and-conquer approach. To find an unknown key, all possible keys are partitioned into a set of groups. This is possible because AES subkeys only have small differences between all rounds. One can then perform a smaller search for the full key because partial bits of the key can then be reused in later phases of the computation thus enhancing the efficiency of computation. In this section we give a brief overview of the key concepts used in biclique cryptanalysis.

3.1 Biclique Structure

Let f be a subcipher that maps an internal state S to a ciphertext C under the key K, i.e. $f_K(S) = C$. Suppose f connects 2^d intermediate states $\{S_j\}$ to 2^d ciphertexts $\{C_i\}$ with 2^{2d} keys $\{K[i,j]\}$ where,

$$\{K[i,j]\} = \begin{bmatrix} K[0,0] & \cdots & K[0,2^d-1] \\ \vdots & \vdots & \vdots \\ K[2^d-1,0] & \cdots & K[2^d-1,2^d-1] \end{bmatrix}.$$

The 3-tuple of sets $[\{S_j\}, \{C_i\}, \{K[i,j]\}]$ is called a d-dimensional biclique, if

$$\forall i,j \epsilon \{0,.....,2^d-1\} : C_i = f_{K[i,j]}(S_j).$$

Considering key length of k bits, initially the key space is partitioned into 2^{k-2d} groups of 2^{2d} keys each. Each key in a group can be represented relative to the base key of the group i.e., $K[0,0]$ and two key differences Δ_i^k and ∇_j^k such that: K[i,j] = K[0,0] $\oplus \Delta_i^k \oplus \nabla_j^k$.

The cipher B is considered as a composition of three parts, $B = f \circ g \circ h$, where h is the subcipher that maps a plaintext P to an internal state V, g is the subcipher that maps V to another internal

state S and f maps internal state S to a ciphertext C i.e., $P \xrightarrow{h} V \xrightarrow{g} S \xrightarrow{f} C$. Once a biclique is constructed for an arbitrary part of the cipher, meet-in-the middle attack is used for the remaining part to recover the key.

3.2 Independent Bicliques

In this paper we use independent biclique construction to attack AES. For each group we choose a base computation i.e., $S_0 \xrightarrow[f]{K[0,0]} C_0$. Then C_i and S_j are obtained using 2^d forward differentials Δ_i i.e. $S_0 \xrightarrow[f]{K[0,0]\oplus\Delta_i^k} C_i$ and 2^d backward differentials ∇_j i.e. $S_j \xleftarrow[f^{-1}]{K[0,0]\oplus\nabla_j^k} C_0$. If the above two differentials do not share active nonlinear components for all i and j, then the following relation is satisfied [11]:

$$S_0 \oplus \nabla_j \xrightarrow[f]{K[0,0]\oplus\Delta_i^k\oplus\nabla_j^k} C_0 \oplus \Delta_i.$$

3.3 Matching with precomputations

In this approach, firstly, the adversary precomputes and stores in memory 2^{d+1} full computations upto a matching state v:

$$\forall i, P_i \xrightarrow{K[i,0]} \overrightarrow{v} \quad \text{and} \quad \forall j, \overleftarrow{v} \xleftarrow{K[0,j]} S_j.$$

Apart from this, the adversary also precomputes and stores $K[i,0]$ and $K[0,j]$ values for the entire Δ_i and ∇_j trails and not till state v. Since in a group, $K[i,0]$ and $K[i,j]$ values do not change much except for the parts affected by i (same holds true for $K[0,j]$ and $K[i,j]$), in particular for i and j, the adversary checks the matching at v by recomputing only those parts of the cipher which differ from the stored ones. In general, matching is done only in a part of state i.e. a single byte v to reduce memory storage and computations.

3.4 Key Recovery

For each group of keys the adversary builds a structure of 2^d plaintexts P_i which map to 2^d intermediate states S_j with respect to the 2^{2d} keys. She then obtains plaintext P_i from ciphertexts C_i through the decryption oracle. If a key in a group satisfies the following relation:

$$P_i \xrightarrow[h]{K[i,j]} \overrightarrow{v} = \overleftarrow{v} \xleftarrow[g^{-1}]{K[i,j]} S_j,$$

then she proposes a key candidate. If a right key is not found in the chosen group then another group is chosen and the whole process is repeated.

3.5 Complexity Calculation

The full complexity of independent biclique attacks is calculated as:

$$C_{full} = 2^{k-2d}(\underbrace{C_{biclique} + C_{precompute} + C_{recompute} + C_{falsepos}}_{Time-Complexity} + \underbrace{C_{decrypt}}_{Data-Complexity}),$$

where,

- $C_{biclique}$ is the computation cost for constructing a biclique.
- $C_{precompute}$ is the cost complexity for calculating v for 2^{d+1}.
- $C_{recompute}$ is the cost complexity of recomputing v for 2^{2d} times.
- $C_{falsepos}$ is the complexity to eliminate false positives.
- $C_{decrypt}$ is the data complexity of oracle to decrypt 2^d ciphertexts.

As mentioned in [11], the full key recovery complexity is dominated by $2^{k-2d} \times C_{recomp}$.

4 Modified Biclique Attack on Full AES-128

In this section we describe the independent biclique construction on full AES-128. We are able to achieve a data complexity of 2^{24} ciphertexts and computation cost of $2^{126.5}$ with this construction as against 2^{88} data complexity and $2^{126.1}$ time complexity described in the previous biclique attack in [11]. The slight increase in computational costs is attributed to small increase in active S-boxes in \$ 0 subkey (10 active boxes compared to 9 boxes in [11]) which further affect the active S-boxes needed to be recomputed in later stages of the differential trail and also due to the change in the biclique trail itself.

4.1 Key Space Partitioning

We divide the 128-bit key space into 2^{112} groups with 2^{16} keys in each group. The ninth round subkey (\$ 9) is taken as the base key ($K[0,0]$) with 2 bytes (16 bits) fixed to 0 and remaining 14 bytes (112 bits) taking all other possible values. Precisely, we take $K[0,0] = (* \; 0 \; * \; * \mid * \; * \; * \; * \mid * \; * \; * \; * \mid 0 \; * \; * \; *)$. Each 2^{112} possible bit combinations define the unique base key for 2^{112} possible groups respectively. All other subkeys can be uniquely determined by the ninth round subkey. The 2^{16} keys ($K[i,j]$) in each group (with respect to base key) are derived using two related key differentials - Δ_i^k and ∇_j^k as follows:

$$K[i,j] = K[0,0] \oplus \Delta_i^k \oplus \nabla_j^k \quad (0 \leq i,j \leq 2^8 - 1),$$

where, Δ_i^k = (0 0 0 0 | 0 0 0 0 | i 0 0 0 | i 0 0 0) and ∇_j^k = (0 j 0 0 | 0 0 0 0 | 0 j 0 0 | 0 0 0 0). It is to be noted that Δ_i^k and ∇_j^k are not differences but actual numerical values.

4.2 2-Round Biclique of Dimension 8

We construct a biclique over the last two rounds (round 9 and 10) of AES-128 as shown in Fig. 1. Due to sliding down, on a change of two bytes in base key (\$ 9), Δ_i differential affects only 6 bytes of the ciphertext C_i with the rest 10 bytes having the same value. Further, $\Delta_i^K(\$10_3) = \Delta_i^K(\$10_7) = \Delta_i^K(\$10_{11}) = \Delta_i^K(\$10_{15})$, hence ciphertext bytes $C_{3,7,11,15}$ always share the same value i.e., C_i = (0 0 0 m | 0 0 0 m | p 0 0 m | q 0 0 m) where $(0 \leq p,q,m \leq 2^8 - 1)$.

4.3 Matching over 8 rounds

We apply matching with precomputations over the remaining rounds of the cipher. (Fig. 2 and Fig. 3) illustrate the matching part for rounds 1-3 in the forward direction and rounds 4-8 in the backward direction. Matching is done on the 0th byte of state v after round 3. The full recomputations required is depicted in Fig. 11. The difference in computation between $P_i \xrightarrow{K[i,0]} \overrightarrow{v}$ and $P_i \xrightarrow{K[i,j]} \overrightarrow{v}$ is determined by the influence of differences between $K[i,0]$ and $K[i,j]$. Similarly, difference in computation between $\overleftarrow{v} \xleftarrow{K[0,j]} S_j$ and $\overleftarrow{v} \xleftarrow{K[i,j]} S_j$ is determined by the influence of differences between $K[0,j]$ and $K[i,j]$. E.g. in Fig. 11, consider $\$8_i$ (violet-colored i.e. $K[8,0]$) and $\$8_j$ (yellow-colored i.e. $K[0,8]$) subkeys. If we map these two on one another we get $K[i,j]$ i.e. $K[8,8]$. Now the difference between $K[0,8]$ and $K[8,8]$ i.e. $K[8,0]$ is reflected upon state #14 (8th byte) of the cipher. Similarly we can find the other parts to be recomputed.

In the forward part from P_i to $\overrightarrow{v}$ there are 10+16+4+1 = 31 active S-boxes in the states. In the backward part from $\overleftarrow{v} \xleftarrow[g^{-1}]{K[i,j]} S_j$ there are 4+16+16+4 = 40 active S-boxes. Since AES key schedule uses only 4 S-boxes per round [1] we also need to calculate active S-boxes in them. There are two active S-boxes in key schedule in Fig. 11($\$1_{12}$ and $\$2_{15}$) that need to be recomputed. Hence in total, there are 73 S-Boxes that need to be recomputed.

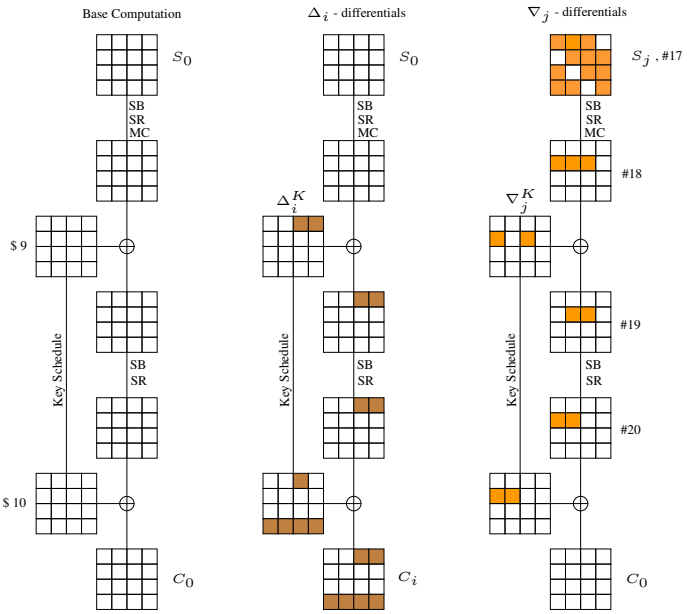

Fig. 1. 2-round Biclique for AES-128

4.4 Complexity Calculations

As discussed in [11], C_{recomp} complexity calculation is majorly driven by the SubBytes operation (as Mix Columns and XOR operation do not contribute much compared to the SubByte operation), hence we calculate C_{recomp} based on total active S-boxes in the SubByte operation only. In AES-128, S-box operation is applied 16 times in each of the 10 state rounds. Hence in total we have 160 S-boxes. Similarly in each key schedule round, S-box is applied 4 times i.e. $10 \times 4 = 40$ S-boxes. As each group has 2^{16} keys, therefore, for each group $C_{recomp} = 2^{16} \times \frac{73}{200} = 2^{14.5}$. Since we match 1 byte i.e. 8 bits in v, we have 2^8 false positives on an average. Similarly $C_{biclique} = 2^{6.68} \approx 2^9 \times \frac{2}{10}$ and $C_{precomp} = 2^{7.68}$. Hence, total running complexity is:

$$C_{time-complexity} = 2^{112} \times (2^{6.68} + 2^{7.68} + 2^{14.5} + 2^8) = 2^{126.5}.$$

As mentioned in Sec. 4.2, in all 2^{112} groups C_i's only differ in 6 bytes (Fig. 1) out of which 4 bytes have the same value (m), hence effectively C_i's differ only in 3 bytes (i.e., p, q and m). Rest 10 bytes have the same fixed value i.e., all 0's. As a result data complexity does not exceed 2^{24} ciphertexts which is significantly lesser than 2^{88} ciphertexts required for previous biclique analysis in [11].The memory required to store one biclique is 2^9 blocks of ciphertexts and intermediate states. The memory requirement for storing precomputations needs 2^8 full computations of $(P_i \xrightarrow{K[i,0]} \overrightarrow{v})$ for 3 rounds and $(\overleftarrow{v} \xleftarrow{K[0,j]} S_j)$ for 5 rounds (Fig. 11).

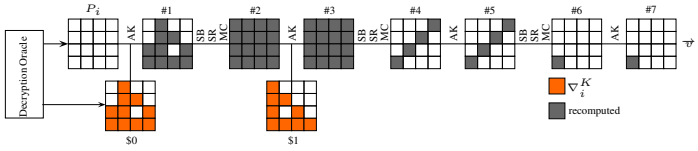

Fig. 2. Forward Recomputation in AES-128.

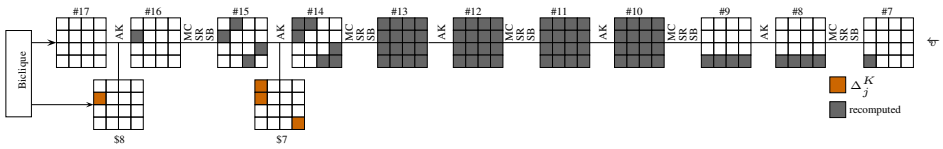

Fig. 3. Backward Recomputation in AES-128.

5 Modified Biclique Attack on Full AES-192

In this section we describe independent biclique attack on full AES-192. The key trail and Δ_i and ∇_j trail gets completely modified due to the sliding down. We are able to achieve a data complexity of 2^{32} ciphertexts and computation costs of $2^{190.5}$ with this construction. The increase in computational costs as compared to [11] is due to change in number and position of active S-boxes in \$ 0 subkey which considerably affects the active boxes needed to be recomputed in later stages of the differential trail and also due to change in biclique trail. The new differential trail is shown in Fig. 5. We also note a few errors in the complexity calculations for this case in [11] which we describe at the end of this section.

5.1 Key Space Partitioning

We divide the 192-bit key space into 2^{176} key groups with 2^{16} keys in each group. A part of tenth round subkey and full eleventh round subkey i.e. ($\$10_R||\$11$) are together taken as the base key $K[0,0]$ with 2 bytes fixed to 0 while the remaining 22 bytes are allowed to take all possible values. Precisely, $K[0,0] = (* * * * \mid * * * * \mid * * * * \mid * * * * \mid * 0 * 0 \mid * * * *)$. All other subkeys can be uniquely determined by $\$10_R||\$11$. The 2^{16} keys $K[i,j]$ (with respect to base key) are derived as:

$$K[i,j] = K[0,0] \oplus \Delta_i^k \oplus \nabla_j^k \quad (0 \leq i,j \leq 2^8 - 1),$$

where, Δ_i^k = (0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 i_1 i_2 0) and ∇_j^k = (0 0 0 0 | 0 0 0 0 | 0 0 0 j | 0 0 0 0 | 0 0 0 j | 0 0 0 0). Δ_i^k defines a tuple (i_1,i_2) where i_1 and i_2 take all possible values ($0 \leq i_1$, $i_2 \leq 2^8$-1)such that after applying the inverse MixColumn operation, it yields a single difference having value i.[1] It is to be noted that Δ_i^k and ∇_j^k are not differences but actual numerical values.

5.2 3-Round Biclique of Dimension 8

We construct a biclique of dimension 8 over the last three rounds (rounds 10-12) as shown in Fig. 4. Due to the slide transition, on a change of 2 bytes in base key ($\$10_R||\$11$), the differential Δ_i affects only 4 bytes of the ciphertext C_i while the rest of the bytes share the same value i.e., C_i = (0 0 0 0 | 0 0 0 0 | 0 i_1 x 0 | 0 y i_2 0) where ($0 \leq i_1, i_2, x, y \leq 2^8 - 1$). There is complete change in Δ_i^k and ∇_j^k key trail due to which Δ_i and ∇_j trail significantly changes impacting a considerable reduction in data complexity (see Fig. 12 in the Appendix for the full trail).

5.3 Matching over 9 rounds

Fig. 5 and Fig. 6 illustrate the matching part. The full recomputations required is depicted in Fig. 12 in the Appendix. There are 4+13+4+1= 22 active S-boxes in the forward part from P_i to $\overrightarrow{v}$. On the other hand, there are 4+16+16+16+6 = 58 active S-boxes in the backward part from $\overleftarrow{v} \xleftarrow[g^{-1}]{K[i,j]} S_j$. There are no active S-boxes in the key schedule. Hence, overall there are 80 S-boxes which need to be recomputed.

[1] It can be seen that there is a one-to-one mapping between i and (i_1,i_2) i.e. for 2^8 i's, we have 2^8 different values of the tuple (i_1,i_2).

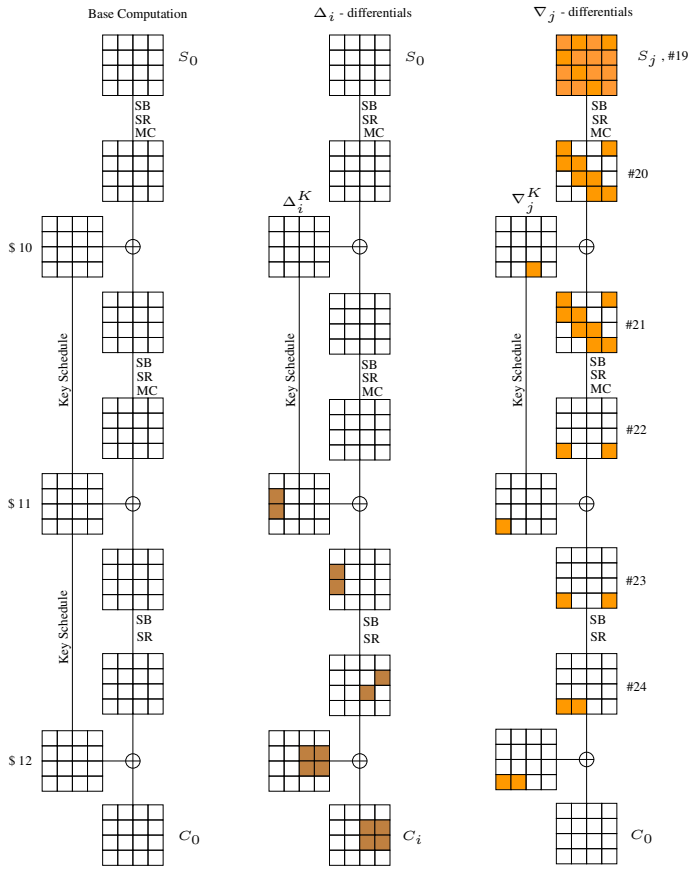

Fig. 4. 3-round Biclique for AES-192

5.4 Complexity Calculations

In AES-192, one full one encryption requires $12 \times 16 = 192$ (state rounds) and $8 \times 4 = 32$ (key rounds) S-boxes. Hence, in total we have 224 S-boxes. For each group, we have $C_{recomp} = 2^{16} \times \frac{80}{224} = 2^{14.5}$, $C_{biclique} = 2^7$, $C_{precomp} = 2^{7.58}$ and $C_{falsepos} = 2^8$. Hence, total running complexity is:

$$C_{time-complexity} = 2^{176} \times (2^7 + 2^{7.58} + 2^{14.5} + 2^8) = 2^{190.5}.$$

As mentioned in Sec. 5.2, in all 2^{176} groups C_i's differ only in 4 bytes (x , y , i_1 , i_2). Rest 12 bytes have the same fixed value i.e., all 0's (Fig. 4). Hence, data complexity does not exceed 2^{32} ciphertexts. The memory required to store one biclique is 2^9 blocks of ciphertexts and intermediate states. The memory requirement for storing precomputations needs 2^8 full computations of $(P_i \xrightarrow{K[i,0]} \overrightarrow{v})$ for 3 rounds and $(\overleftarrow{v} \xleftarrow{K[0,j]} S_j)$ for 6 rounds (see appendix Fig. 12).

Discrepancies: We would like to point here a calculation mistake in [11]. The required S-box calculation is given as 2.8125 Sub-Bytes operations (45 S-boxes operations) i.e. $2^{13.68}$ runs of full AES-192 whereas it should be 3.8125 Sub Byte operations (61 S-boxes) i.e. $2^{14.13}$ runs of full AES-192. As a result, the full computational complexity in [11] should be $2^{176} \times 2^{14.13} = \mathbf{2^{190.13}}$.

6 Modified Biclique Attack on Full AES-256

In this section we describe independent biclique attack on full AES-256. Due to sliding down of the biclique, we get a data complexity of just 2^8 (as compared to 2^{40} in [11]) with time complexity of

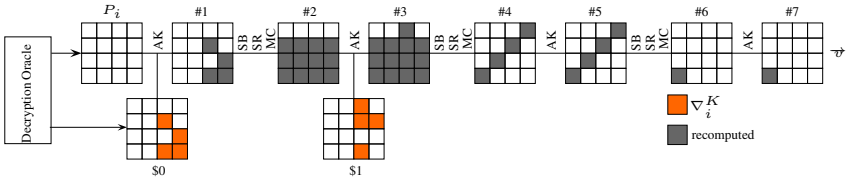

Fig. 5. Forward Recomputation in AES-192.

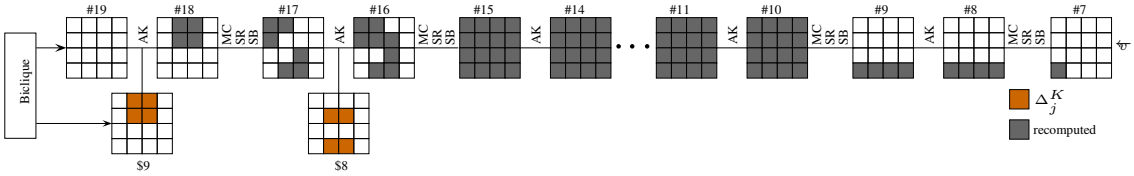

Fig. 6. Backward Recomputation in AES-192.

$2^{254.72}$ (as compared to $2^{254.42}$ reported in [11]). The increase in computation cost is attributed to change in the number and position of active S-boxes in $ 0 subkey which further affects the active boxes needed to be recomputed in the later stages of the differential trail (shown in Fig. 8 in the Appendix) and and also due to changes in the biclique trail. We also find an errror in the differential trail and complexity calculations described in [11] which we explain at the end of this section.

6.1 Key Space Partitioning

We divide the 256-bit key space into 2^{240} key groups with 2^{16} keys in each group. Twelveth and thirteenth round subkeys i.e. ($12||$13) are together taken as the base key $K[0,0]$ with 2 bytes fixed to 0 while the remaining 30 bytes are allowed to take all possible values. Precisely, $K[0,0]$ = (* * * * | * * * * | * 0 * * | * * * * | * * 0 * | * * * * | * * * * | * * * *). All other subkeys can be uniquely determined by $12||$13. The 2^{16} keys $K[i,j]$ are derived as:

$$K[i,j] = K[0,0] \oplus \Delta_i^k \oplus \nabla_j^k \quad (0 \leq i,j \leq 2^8 - 1),$$

where Δ_i^k = (0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 i 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0) and ∇_j^k = (0 0 0 0 | 0 0 0 0 | 0 j 0 0 | 0 j 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0). It is to be noted that Δ_i^k and ∇_j^k are not differences but actual numerical values.

6.2 3-Round Biclique of Dimension 8

We construct a biclique of dimension 8 over the last three rounds (rounds 12-14). The biclique is shown in Fig. 7. Due to the sliding down of the biclique, on a change of 1 byte in base key ($12||$13) the differential Δ_i affects only 1 byte of the ciphertext C_i while rest of the bytes share the same value i.e., C_i = (0 0 0 0 | 0 0 0 0 | 0 0 r 0 | 0 0 0 0) where (0 ≤ r ≤ 2^8-1).

6.3 Matching over 11 rounds

Fig. 8 and Fig. 9 illustrate the matching part. The full recomputations required is depicted in Fig. 13 in the Appendix. There are 5+13+4+1= 23 active S-boxes in the forward part from P_i to $\overrightarrow{v}$. On the other hand, there are 4+16*5+8 = 92 active S-boxes in the backward part from $\overleftarrow{v} \xleftarrow{K[i,j]} S_j$. There are no active S-boxes in key schedule. Hence, overall there are 115 S-boxes which need to be recomputed.

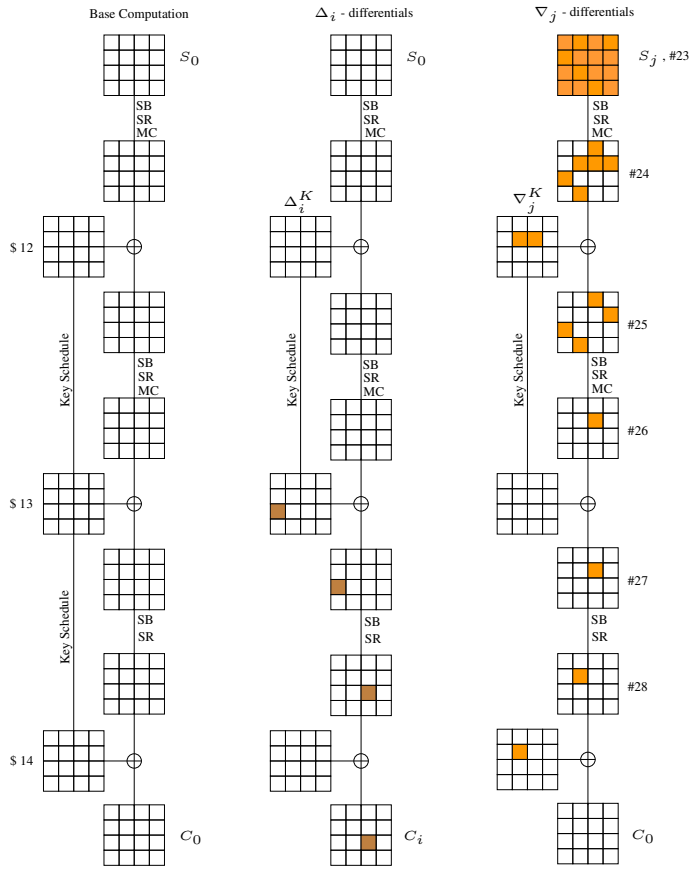

Fig. 7. 3-round Biclique for AES-256.

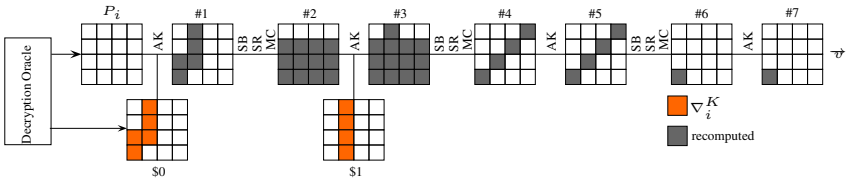

Fig. 8. Forward Recomputation in AES-256.

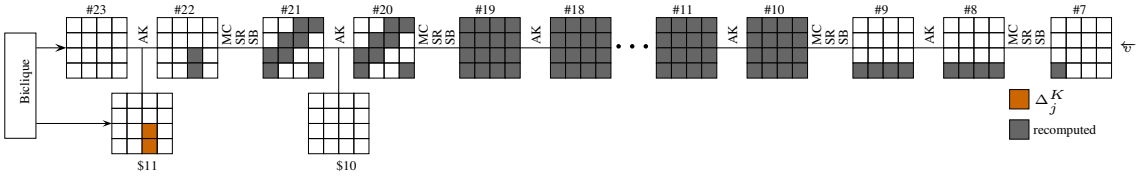

Fig. 9. Backward Recomputation in AES-256.

6.4 Complexity Calculations

One full encryption in AES-256 requires $14 \times 16 = 224$ (state rounds) and $6 \times 8 + 4 = 52$ (key rounds) S-boxes. Hence in total we have 276 S-boxes. For each group, we have $C_{recomp} = 2^{16} \times \frac{115}{276} = 2^{14.75}$, $C_{biclique} = 2^{6.98}$, $C_{precomp} = 2^{8.65}$ and $C_{falsepos} = 2^8$. Hence, total running complexity is:

$$C_{time-complexity} = 2^{240} \times (2^{6.98} + 2^{8.65} + 2^{14.75} + 2^8) = 2^{254.72}.$$

As discussed in Sec. 6.2, in all 2^{240} groups C_i's differ only in 1 byte (r). Rest all 15 bytes have the same fixed value i.e., all 0's. Hence, the data complexity does not exceed 2^8. The memory required to store one biclique is 2^9 blocks of ciphertexts and intermediate states. The memory requirement for storing precomputations needs 2^8 full computations of $(P_i \xrightarrow{K[i,0]} \overrightarrow{v})$ for 3 rounds and $(\overleftarrow{v} \xleftarrow{K[0,j]} S_j)$ for 8 rounds (see appendix Fig. 13).

Discrepancies: We would like to point certain discrepancies in [11]. Firstly, in their Fig. 12, \$ 0 and \$ 1 subkeys have been wrongly marked as \$1 and \$2 subkeys respectively. Secondly, the authors in [11] state that whitening subkeys differ in 1 byte only, whereas they differ in 4 bytes. Thirdly, as a consequence of the second discrepancy the number of active S-boxes shown in #1, #2 and #3 in their Fig. 12 are less. According to our calculations, the correct trail should be as follows:

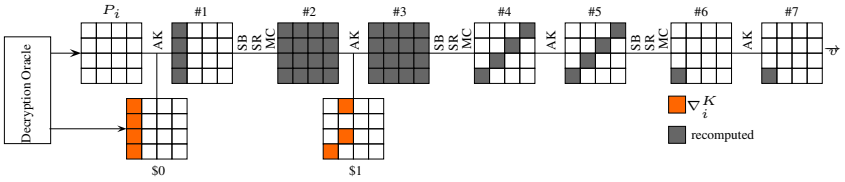

Fig. 10. Corrected AES-256 forward computation.

Hence the required S-box calculation should be 6.3125 Sub-Bytes operation (101 S-boxes) i.e. 14.6 runs of full AES-192 as against 5.4375 Sub-Bytes operations (87 S-boxes operations)[2]. As a result, the full computational complexity in [11] should be $2^{240} \times 2^{14.6} = \mathbf{2^{254.6}}$.

7 Conclusion and Discussion

In this work, we have attempted to overcome the inherent disadvantage of very high query complexity associated with the technique of biclique cryptanalysis when applied to AES. We show that instead of solely focusing on maximizing the rounds over which a biclique can be constructed in order to reduce the key recovery effort to minimum, a better approach is to construct bicliques such that a good balance is obtained between computational complexity and data complexity. Through our new biclique construction, we could bring down the oracle queries needed to decrypt the ciphertexts to a significantly lower limit which can be achieved by an attacker having reasonable computing power within reasonable period of time. We agree that computational costs do slightly increase in this construction but when compared to previous attack, the increase is very diminutive making our biclique construction more viable for practical implementations. Reducing both data complexity and the query complexity while using the technique of biclique cryptanalysis seems difficult and is an interesting open problem.

[2] [11] reports two different values of total active S-boxes (5.4375 and 5.625) in the same paragraph.

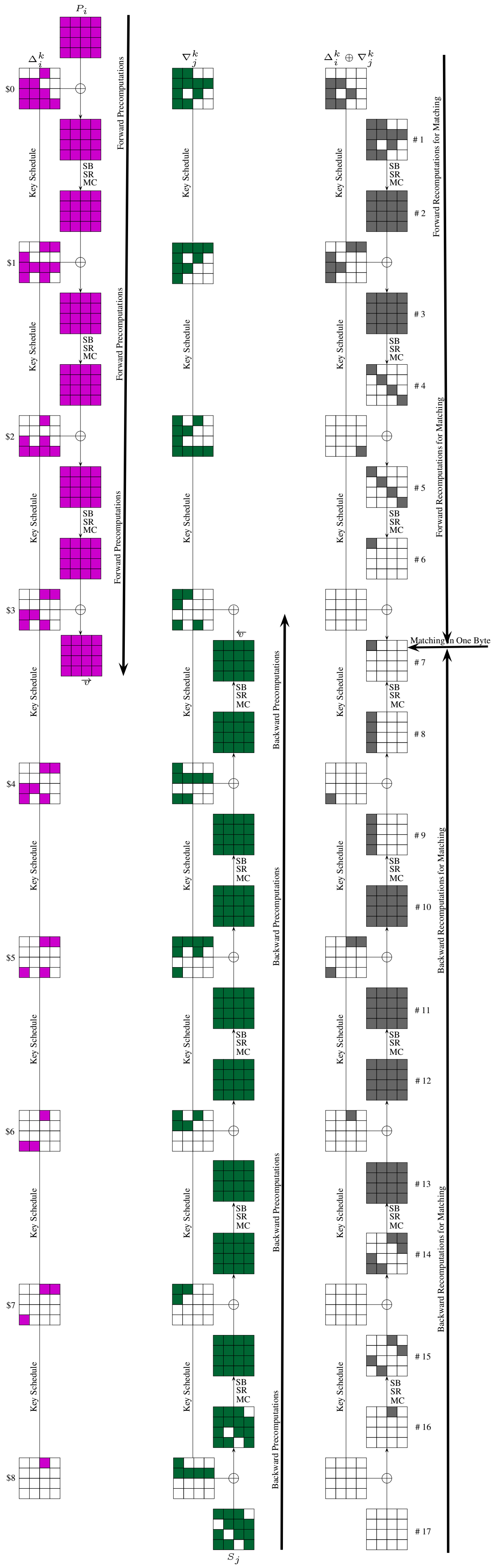

Fig. 11. Precomputations and Recomputations required in AES-128. The violet and green colored boxes show the precomputations required in Δ_i trail and ∇_j trail respectively. The gray colored boxes show the recomputations required in $\Delta_i \oplus \nabla_j$ for matching.

References

1. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002.
2. Michael Gorski and Stefan Lucks. New Related-Key Boomerang Attacks on AES. In *Indocrypt 2008* [18], pages 266–278.
3. Alex Biryukov. The Boomerang Attack on 5 and 6-Round Reduced AES. In *AES Conference*, volume 3373 of *Lecture Notes in Computer Science*, pages 11–15. Springer, 2004.
4. Raphael Chung-Wei Phan. Impossible differential cryptanalysis of 7-round Advanced Encryption Standard (AES). *Information Processing Letters*, 91(1):33–38, 2004.
5. Jiqiang Lu, Orr Dunkelman, Nathan Keller, and Jongsung Kim. New Impossible Differential Attacks on AES. In *Indocrypt 2008* [18], pages 279–293.
6. Niels Ferguson, Richard Schroeppel, and Doug Whiting. A simple algebraic representation of rijndael. In *Selected Areas in Cryptography 2001*, volume 2259 of *Lecture Notes in Computer Science*, pages 103–111. Springer, 2001.
7. Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Asiacrypt 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer, 2002.
8. Alex Biryukov and Dmitry Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In *Asiacrypt 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2009.
9. Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. Distinguisher and Related-Key Attack on the Full AES-256. In *Crypto 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, 2009.
10. Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key Recovery Attacks of Practical Complexity on AES-256 Variants with up to 10 Rounds. In *Eurocrypt 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer, 2010.
11. Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique Cryptanalysis of the Full AES. In *Asiacrypt 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 344–371. Springer, 2011.
12. Dmitry Khovratovich, Christian Rechberger, and Alexandra Savelieva. Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. In *Fast Software Encryption 2012*, volume 7549 of *Lecture Notes in Computer Science*, pages 244–263. Springer, 2012.
13. Shao zhen Chen and Tian min Xu. Biclique Attack of the Full ARIA-256. *IACR Cryptology ePrint Archive*, 2012:11, 2012.
14. Hamid Mala. Biclique Cryptanalysis of the Block Cipher SQUARE. *IACR Cryptology ePrint Archive*, 2011:500, 2011.
15. Mustafa Çoban, Ferhat Karakoç, and Özkan Boztas. Biclique Cryptanalysis of TWINE. *IACR Cryptology ePrint Archive*, 2012:422, 2012.
16. Deukjo Hong, Bonwook Koo, and Daesung Kwon. Biclique Attack on the Full HIGHT. In *ICISC 2011*, volume 7259 of *Lecture Notes in Computer Science*, pages 365–374. Springer, 2011.
17. Farzaneh Abed, Christian Forler, Eik List, Stefan Lucks, and Jakob Wenzel. Biclique Cryptanalysis of the PRESENT and LED Lightweight Ciphers. *IACR Cryptology ePrint Archive*, 2012:591, 2012.
18. *Progress in Cryptology - INDOCRYPT 2008, 9th International Conference on Cryptology in India, Kharagpur, India, December 14-17, 2008. Proceedings*, volume 5365 of *Lecture Notes in Computer Science*. Springer, 2008.

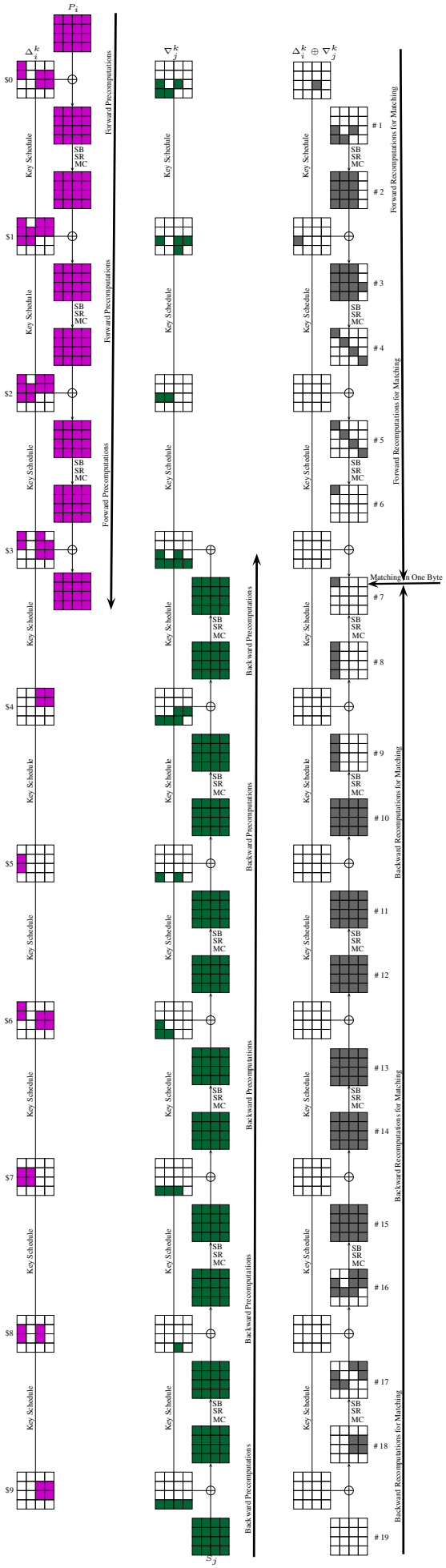

Fig. 12. Precomputations and Recomputations required in AES-192. The violet and green colored boxes show the precomputations required in Δ_i trail and ∇_j trail respectively. The gray colored boxes show the recomputations required in $\Delta_i \oplus \nabla_j$ for matching.

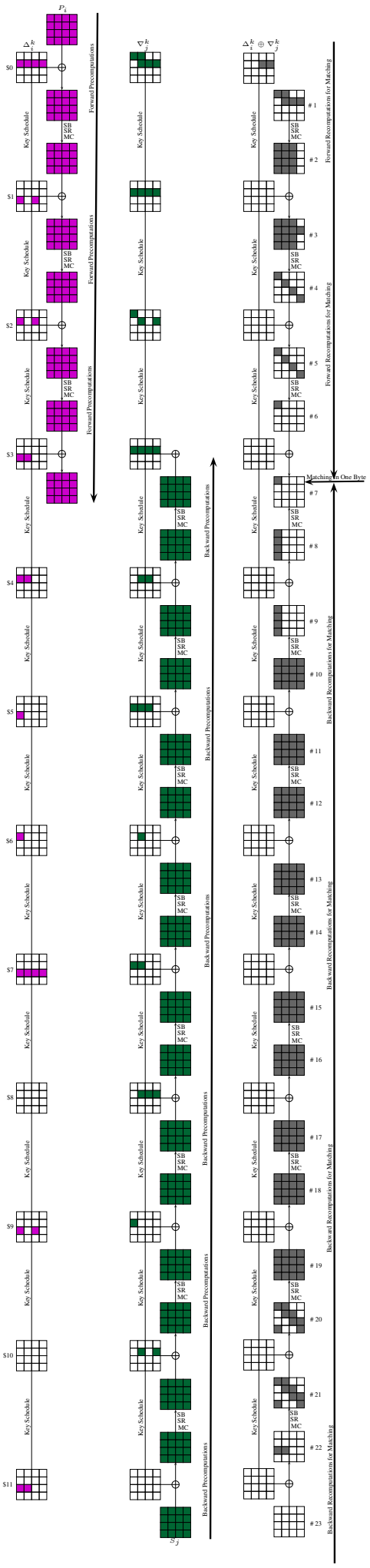

Fig. 13. Precomputations and Recomputations required in AES-256. The violet and green colored boxes show the precomputations required in Δ_i trail and ∇_j trail respectively. The gray colored boxes show the recomputations required in $\Delta_i \oplus \nabla_j$ for matching.