Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2024) 32: 338 – 357
© TÜBİTAK
doi:10.55730/1300-0632.4072

Lower data attacks on Advanced Encryption Standard

Orhun KARA[1,2*]
[1]Department of Mathematics, Faculty of Science, İzmir Institute of Technology (IZTECH), İzmir, Turkiye
[2]TÜBİTAK BİLGEM UEKAE Gebze, Kocaeli, Turkiye

Received: 25.11.2023 • **Accepted/Published Online:** 08.02.2024 • **Final Version:** 14.03.2024

Abstract: The Advanced Encryption Standard (AES) is one of the most commonly used and analyzed encryption algorithms. In this work, we present new combinations of some prominent attacks on AES, achieving new records in data requirements among attacks, utilizing only 2^4 and 2^{16} chosen plaintexts (CP) for 6-round and 7-round AES-192/256, respectively. One of our attacks is a combination of a meet-in-the-middle (MiTM) attack with a square attack mounted on 6-round AES-192/256 while another attack combines an MiTM attack and an integral attack, utilizing key space partitioning technique, on 7-round AES-192/256. Moreover, we illustrate that impossible differential (ID) attacks can be viewed as the dual of MiTM attacks in certain aspects which enables us to recover the correct key using the meet-in-the-middle (MiTM) technique instead of sieving through all potential wrong keys in our ID attack. Furthermore, we introduce the constant guessing technique in the inner rounds which significantly reduces the number of key bytes to be searched. The time and memory complexities of our attacks remain marginal.

Key words: Block cipher, Advanced Encryption Standard, meet-in-the-middle attack, square attack, cryptanalysis, encryption

1. Introduction

AES, as defined by the National Institute of Standards and Technology (NIST) [1], stands as a prominent block cipher extensively deployed for ensuring confidentiality in various cryptographic protocols. These protocols include, but are not limited to, wireless security, file and database encryptions, Transport Layer Security (TLS), GSM-5G, WiFi Protected Access (WPA), and the Signal protocol integrated into ubiquitous applications such as WhatsApp. Therefore, any analysis of AES within specific parameters, particularly scenarios involving limited data, assumes a critical role. Such analyses play a vital role in enhancing our understanding of the security implications associated with commonly employed ciphers, facilitating a comprehensive evaluation of their security against attacks using a practical amount of data.

AES stands as one of the most extensively cryptanalyzed ciphers, with numerous attack techniques mounted on its reduced rounds across distinct key lengths. This substantial body of work significantly contributes to the cryptanalysis of block ciphers. Noteworthy analyses encompass Meet-in-The-Middle (MiTM) attacks, such as those by Demirci and Selçuk [2], Dunkelman et al. [3, 4], Wang and Zhu [5], Derbez et al. [6], Li et al. [7], Gilbert and Minier [8], square attacks [9], biclique attacks [10, 11], yoyo attacks [12, 13], truncated boomerang attacks [14], zero difference attacks [15], algebraic attacks [16], mixture differential attacks [17], mixture integral attacks [18], and impossible differential (ID) attacks [19–32]. Additionally, the key schedule of

*Correspondence: orhunkara@iyte.edu.tr

AES has been subjected to intensive cryptanalysis [4, 33]. ID attacks, initially discovered by [34, 35], exploit impossible inner differences and usually require too much data to sieve all the wrong keys in the encryption and decryption directions.

The feasibility of the best attacks on block ciphers is often limited due to their substantial data requirements. This is attributed to the inherent challenge that if the data complexity of an attack is practically low, there usually exists the potential for decreasing the time complexity by increasing the data complexity. A typical illustration is found in the case of the Data Encryption Standard (DES) cipher. Both the differential attack [36] and the linear attack [37] are much faster than the exhaustive search. Nevertheless, the brute-force attack remains the most practical means of recovering a DES key, as these attacks require several terabytes of data. Consequently, the significance of low-data attacks becomes particularly vital when evaluating the security level of a cipher. Similar security analyses are conducted on AES to understand the security implications of reduced rounds when only a practical amount of data is available. These investigations are particularly significant since they serve as a benchmark against exhaustive search methods, offering insights into how far AES's security deviates from the expected level based on its key length because brute force attacks fall into the category of low-data complexity attacks. However, their time complexity can be impractically high. Therefore, a crucial aspect of analyzing reduced rounds of AES involves studying the minimum data requirements, with considerations of time and memory complexities as secondary issues [9, 12, 38–43].

Table 1. Low data attacks on AES with 6 and 7 rounds. D, T, and M stands for data (in CP), time, and memory (in byte) complexities, respectively.

Variant	D	T/M	Round	Reference
All	2^{26}	2^{80} / 2^{35}	6	[38]
AES-192	2^{18}	$2^{180}/2^{78}$	6	[44]
AES-256	2^{18}	2^{186} / 2^{43}	6	[44]
AES-192	16	$2^{146}/2^{153}$	6	Section 6
AES-256	16	$2^{163}/2^{169}$	6	Section 6
AES-192	2^{26}	$2^{153}/2^{32}$	7	[40]
AES-192/256	2^{26}	$2^{146.3}/2^{40}$	7	[40]
AES-192	2^{16}	$2^{171}/2^{154}$	7	Section 7
AES-256	2^{16}	$2^{173}/2^{170}$	7	Section 7

The challenge of determining the minimum data requirements for attacks on 4 and 5 rounds of AES is nearly resolved. Bouillaguet et al. proposed an attack with a time complexity of 2^{104} on 4-round AES, utilizing only 2 chosen plaintexts (CP). Their attack remains within practical time limits even when using 4 CP, requiring only 2^{32} AES encryptions [39]. For attacks on 5-round AES, the minimum data requirement is established at 8 CP, with a time complexity of 2^{64} [45].

It is apparent that identifying the lowest data complexity among attacks on AES with more than 5 rounds presents a significant challenge, unlike in the cases of 4 and 5 rounds. The square attack, requiring 2^{32} chosen plaintexts for 6-round AES, maintained its record for nearly two decades [46].

Despite subsequent improvements, including the technique for partial summing in the square attack [9] and advanced MiTM attacks [5, 6], achieving superior time complexities, none have surpassed the data complexity established by the square attack in [46].

Table 2. The best attacks on 6-round AES and 7-round AES. They require 2^{33} and 2^{97} data, respectively. D, T, and M stand for data (in CP), time, and memory (in byte) complexities, respectively. *: Complexity is given as the number of additions in FFT.

Variant	D	T	M	Round	Reference
All	2^{33}	2^{44}	2^{37}	6	[44]
All	$6 \cdot 2^{32}$	2^{46}	$6 \cdot 2^{36}$	6	[9]
All	2^{33}	$2^{46.4*}$	2^{31}	6	[47]
All	2^{97}	2^{99}	2^{98}	7	[6]
AES-128	2^{106}	2^{110}	2^{90}	7	[26]

Significant improvements have been made in Meet-in-the-Middle (MiTM) techniques applied to AES since the pioneering Demirci-Selçuk attack [2]. Notably, after 18 years, Bar-On et al. made significant strides by breaking the record, achieving $2^{27.5}$ CP using the mixed MiTM technique [40]. Further refinement resulted in a reduced data complexity of 2^{26} CP in [38]. It is essential to emphasize that, despite setting the record for minimum data complexity, this particular attack did not hold the title for the fastest method against 6-round AES. A recent study marked a significant improvement in this record, achieving 2^{18} chosen plaintexts for attacks on 6-round AES with a very high time complexity [44]. As of now, the minimum data requirement for a 7-round AES stands at 2^{26} CP [40].

1.1. Our contributions

In this work, we investigate low-data attacks on both 6-round and 7-round AES and significantly enhance the lowest data requirements. We employ a novel combination of Meet-in-the-Middle (MiTM) and square attacks on 6-round AES, requiring only 16 CP. Our attacks succeed with 192-bit and 256-bit key lengths. Additionally, we apply the key partitioning technique from [48] in our MiTM attack and use identically active sets to improve the lowest data complexity for 7-round AES. This time, we achieve 2^{16} CP for both AES-256 and AES-192. Table 1 provides details of the low-data attacks on 6-round and 7-round AES.

In our differential MiTM attacks, we employ a single structure and fix a few active bytes, leaving the remaining bytes not only passive but also constant. Thus, we utilize the constant guessing technique, significantly reducing the search space. Furthermore, all the bytes do not have to be active for an active column in this case. Essentially, we treat our differentially active bytes as integrally active as well, rendering them like permutations, while the other bytes remain constant. As a result, as shown in Table 1, we achieve a significant improvement in the lowest data complexity.

The paper is organized as follows. We give a brief decryption of AES in Section 2. We introduce our basic attack in Section 3 and the constant guessing technique used in our attacks in Section 4. An improvement of the basic attack is given in Section 5. We present a combination of MiTM attack and square attack to achieve the minimum data in Section 6. The extension of this attack through the key partitioning technique is introduced in Section 7. Finally, we conclude the paper with Section 8.

2. A short definition of AES

AES is the FIPS 197 standard [1]. We give a short definition of AES. A detailed decryption along with test vectors can be found in [1, 41]. It is a block cipher with 128-bit block length. There are three options for the lengths of the key: $k = 128$, 192 or $k = 256$ bits. These lengths correspond to $r = 10, 12$ and $r = 14$ rounds, respectively. It is common to depict 16 bytes of an inner state of AES by a matrix of 4×4 dimensions. There are four operations of AES in one round (see Figure 1):

SubBytes (SB): It consists of 16 8×8 S-box operations. It is a substitution of each byte through a lookup table.

ShiftRows (SR): The SR operation is a cyclic rotation of bytes. The i-th row is rotated $i-1$ byte to the left for $i = 2, 3$ and $i = 4$.

MixColumns (MC): It is a matrix multiplication of each column by an MDS matrix over the extension Galois field $GF(256)$. Each column of the input state is multiplied by this 4×4 MDS matrix and it is substituted with the output.

AddRoundKey (ARK): XORs the j-th byte of the output state of the i-th round with the j-th byte of the i-th subkey.

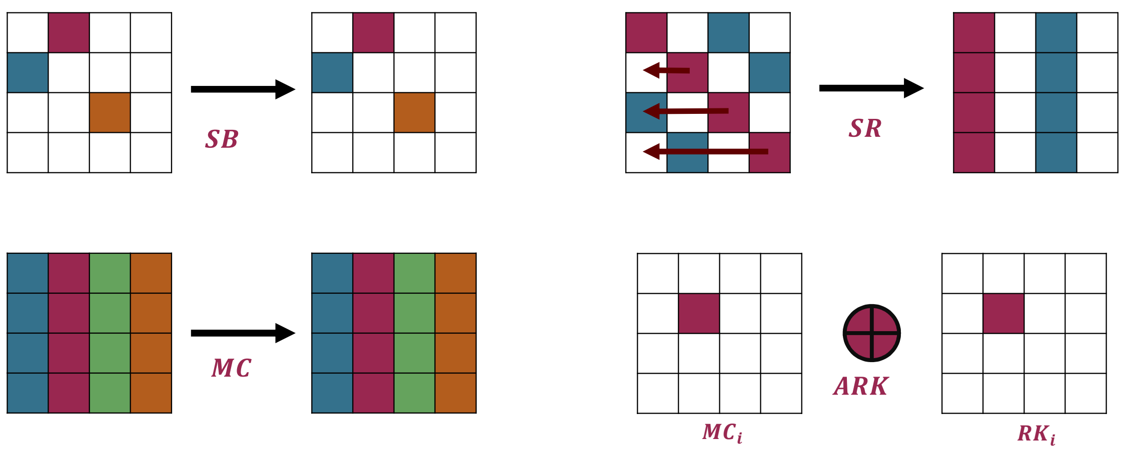

Figure 1. Operations of AES in one round

The first subkey RK_0 is considered the whitening key and it is added to the plaintext prior to the encryption. The last round lacks MC operation. We denote the inverse operations as SB^{-1}, SR^{-1}, and MC^{-1}, which are the inverses of SB, SR, and MC, respectively.

We introduce the key schedule of AES-192 briefly since we only exploit it in our attacks. Let RK_0 be the whitening key. The recursive relation between columns of the subkeys of AES-192 is given as

$$RK\{k\} = \begin{cases} RK\{k-6\} \oplus \phi(RK\{k-1\}) \oplus \mathrm{r}\{k \text{ div } 6\}, & \text{if } (k \bmod 6) = 0, \\ RK\{k-6\} \oplus RK\{k-1\}, & \text{else;} \end{cases} \tag{1}$$

where $RK\{4j+i-1\}$ is the i-th column of the j-th subkey, ϕ is an S-box-based function on columns and $\mathrm{r}\{k\}$s are round constants.

2.1. Notation

Let P, C, K, RK_i, and ΔS denote a plaintext, a ciphertext, a main key, the ith round key, and the difference of a pair for S, respectively. For instance, ΔP is the plaintext differences. We indicate both the output of a round operation and the round number with a subindex. For instance, ΔSR_i stands for the output difference of a pair of data of the SR operation in the ith round. We comply with the same indexing for the inverse functions SB_i^{-1}, MC_i^{-1}, ΔSB_i^{-1}, and ΔMC_i^{-1}. If a specific input or output of these functions must be pointed out, we use $MC_i(X)$, $SB_i(X)$, or $MC_i^{-1}(X)$.

The byte numbers are ordered in the 4×4 matrix as in Figure 2. Thus, bytes with indices $0, 4, 8, 12$ are located in the first column. We denote the byte positions of a state in $[\cdot]$. That is, $T[\alpha_1, \ldots, \alpha_\ell]$ denotes the $(\alpha_1, \ldots, \alpha_\ell)$-th positions of the state T, respectively. For instance, $MC_2[1, 7]$ denotes the second and the 8-th bytes of the output of the MC operation in the second round in the ordering depicted in Figure 2. $\Delta MC_2^{-1}[2, 5]$ means the third and the fifth bytes of a given input difference of the MC operation in the second round.

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Figure 2. Byte numbers of a state.

3. Basic attack on 6-round AES

Our primary attack is an impossible differential (ID) attack on AES. Our approach differs from a conventional ID attack on AES in two key aspects. We exploit a unique ID characteristic and eliminate the restriction that all bytes of an active MC operation must be active when the subsequent SB operations are in use.

The majority of ID attacks on AES in the literature exploit one of the ID characteristics within the family of 4-round characteristics defined by Grassi et al. in [49]. If, in any 4-round characteristic of AES, the sum of active columns after the SR and SR^{-1} operations in the encryption and decryption directions, respectively, is not greater than 4, it defines an ID characteristic [49]. This characteristic is referred to as a conventional ID characteristic of AES [44].

We do not exploit conventional ID characteristics. The contradiction in our characteristic occurs during the SR operation of the fourth round, as depicted in Figure 3. $SR_4[0]$ is active in the encryption direction, whereas it is passive in the decryption direction. This characteristic is key-dependent, relying on the subkeys in both the encryption and decryption directions. Any key candidate that leads to this contradiction is deemed incorrect and, consequently, sieved.

Recovering the key through our basic attack follows the classical steps of a standard ID attack, having two parts. Firstly, we determine the ciphertext pairs for each guess of the round keys that result in passive bytes in $SR_4[0]$ on the decryption side. Later, we utilize the corresponding plaintext pairs on the encryption side, leading to active bytes in $SR_4[0]$ during the encryption process.

The dataset consists of 2^8 CP. In each plaintext, the first byte, $P[0]$, takes all possible values, while the other bytes remain constant. Formally, the dataset is defined as:

$$D = \{P_i : P_i[0] = i \text{ for } i = 0, \dots, 255;\ P_j[z] = P_k[z]\ \forall j, k = 0, \dots, 255 \text{ for } z \neq 0\}.$$

The corresponding ciphertexts are denoted as $C_i = E_K(P_i)$. The total number of pairs is approximately 2^{15}, specifically (P_i, P_j) for $i \neq j$. It is important to note that we only use one structure of such a set.

3.1. Preparing tables in decryption side

Constructing the table is a routine procedure in a classical ID attack, involving guess-and-determine and early abort techniques. Further details can be found in [23]. Thus, we briefly explain how to prepare our table which contains each subkey guess in the decryption direction and the corresponding ciphertext pairs leading to $\Delta MC_4^{-1}[0] = 0$.

First of all, let us guess the whole round key RK_6. We can decrypt each pair (C_i, C_j) for one round and then guess $MC_5^{-1}(RK_5)[0, 7, 13]$ for AES-256. Let us remark that we can compute the first two columns of RK_5, $RK_5\{0, 1\}$, from RK_6 by means of the key schedule for AES-192. Subsequently, we simply compute $MC^{-1}(RK_5)[0, 13]$. Therefore, it is enough to guess only $MC^{-1}(RK_5)[7]$ for AES-192.

Compute $\Delta SB_5^{-1}[0, 4, 12]$. The difference for a pair will be $\Delta MC_4[0, 4, 12]$ since the round key RK_4 does not change the difference. Then we can compute $\Delta MC_4[8]$ from the linear equation $\Delta MC_4^{-1}[0] = 0$ since we want $\Delta SR_4[0] = 0$. On the other hand, $\Delta MC_4[8] = \Delta SB_5^{-1}[8]$, but we know $\Delta SB_5[8]$. The probability that there is a transition for the input/output differences $\Delta SB_5^{-1}[8] \to \Delta SB_5[8]$ through the difference distribution table of SB of AES is around 1/2 and there are most likely 2 solutions. Therefore, we can determine two values of $MC_5^{-1}(RK_5)[10]$ for roughly half of the pairs, as depicted in Figure 3.

The probability that $\Delta MC_4^{-1}[0] = 0$ is 2^{-8}. Hence, we expect approximately $2^7 = 128$ pairs out of 2^{15} pairs for each subkey guess in the decryption side. We have 2^{160} and 2^{136} key candidates for RK_5 and $MC^{-1}(RK_5)[0, 7, 10, 13]$ in AES-256 and AES-192, respectively. Let us store each key candidate with its roughly 128 ciphertext pairs resulting in the equality $\Delta SR_4[0] = 0$. The complexities of preparing these tables are $2 \cdot 2^{14} \cdot 2^{152} = 2^{167}$ and $2 \cdot 2^{14} \cdot 2^{136} = 2^{151}$ two-round decryptions of AES-256 and AES-192, respectively.

4. Constant guesses and partially active MC

We guess constant bytes in the encryption direction to check if $\Delta SR_4[0] \neq 0$ for each plaintext pairs. This will enable us both to decrease the number of bytes we must guess and to overcome the passive bytes of MC_2. All the bytes of an active MC operation are expected to be active in a typical ID attack when there are subsequent active SB operations, as the differences need to be evaluated throughout SB. It is important to note that the decryption part of our ID attack is standard, with four of the MC^{-1} operations being active in the fifth round (see Section 3). Therefore, all bytes must be active, resulting in guessing the entire RK_6. However, a distinction arises in the encryption direction, as in the second round, where all four MC operations are active, but only one byte in each MC is active (see Figure 3).

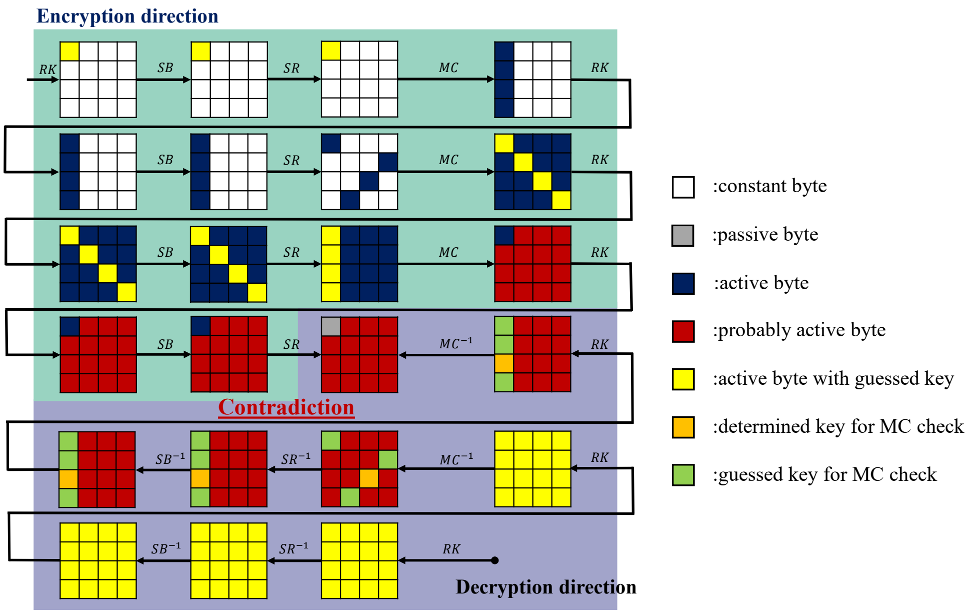

Figure 3. In our basic attack, 2^8 CP whose first bytes take all value and the other bytes are constant, are utilized along with their corresponding ciphertexts. The white boxes represent constant bytes. They do not change with the plaintext. We carefully determine internal state bytes and subkey bytes to be guessed for recovering $\Delta SR_4[0]$, the difference in the first byte of the state in round four after the SR operation, in both encryption and decryption directions. A contradiction arises if it is passive in one direction and active in the other direction.

The passive bytes in the plaintext pairs are also constant. That is, the value in a passive byte do not change across any plaintext. The constant bytes of a plaintext remain constant throughout the operation of AES up to the second round. For any pair P_1, P_2 of plaintexts taken in one structure, we have $\Delta SR_2[1,2,3,4,5,6,8,9,11,12,14,15] = 0$ in a conventional ID attack. However, we use only one structure. Therefore, $\Delta SR_2[1,2,3,4,5,6,8,9,11,12,14,15]$ and any differences in all the other passive bytes in the encryption direction are zero for any plaintext pairs. This implies that all the passive bytes take constant values. Our objective is to identify these constant values rather than subkey bytes. We explain how to exploit this property and introduce the complexities for the basic attack in this section.

The passive bytes of MC_2 are present even when we need to run the SB_3 operation. We must recover $\Delta MC_3[0]$ to check if a pair leads to ID characteristic and we compute this difference by guessing the constant secret values. In fact we are supposed to check if $\Delta SR_4[0] \neq 0$ for the miss-in-the-middle. This condition is equivalent to $\Delta MC_3[0] \neq 0$.

We guess only two subkey bytes: $RK_0[0]$ and $MC_1^{-1}(RK_1)[0]$. All the other guesses are constant state bytes. First of all, we guess three bytes: $SR_1[4,8,12] \oplus MC_1^{-1}(RK_1)[4,8,12]$, which remains constant for any plaintext used. It is possible to recover $SR_2[0,7,10,13]$ by these 2+3=5 byte guesses. These represent the active bytes before MC_2 (see Figure 3). Observe that the following four bytes , $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$, and $\mathcal{C}_4$, are constant and does not change with any plaintext.

$$\begin{aligned}
\mathcal{C}_1 &= 3SR_2[4] \oplus SR_2[8] \oplus SR_2[12] \oplus MC^{-1}(RK_2)[0], \\
\mathcal{C}_2 &= SR_2[1] \oplus 2SR_2[5] \oplus 3SR_2[9] \oplus MC^{-1}(RK_2)[5], \\
\mathcal{C}_3 &= SR_2[2] \oplus SR_2[6] \oplus 3SR_2[14] \oplus MC^{-1}(RK_2)[10], \text{ and} \\
\mathcal{C}_4 &= 3SR_2[3] \oplus SR_2[11] \oplus 2SR_2[15] \oplus MC^{-1}(RK_2)[15].
\end{aligned}$$

Therefore, let us guess four constant bytes; $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$, and $\mathcal{C}_4$. Then, we can compute $SR_3[0] = SB(2 \cdot SR_2[0] \oplus \mathcal{C}_1)$. Similarly, $SR_3[4, 8, 12]$ can be computed through $\mathcal{C}_2, \mathcal{C}_3$, and $\mathcal{C}_4$. Indeed, $SR_3[4] = SB(SR_2[13] \oplus \mathcal{C}_2)$, $SR_3[8] = SB(2 \cdot SR_2[10] \oplus \mathcal{C}_3)$, and $SR_3[12] = SB(SR_2[13] \oplus \mathcal{C}_4)$. After recovering $SR_3[0, 4, 8, 12]$, we can check whether $\Delta MC_3[0] = MC(\Delta SR_3[0, 4, 8, 12])$ is nonzero. Our total guesses are 2+3+4=9 bytes.

The remaining part of the attack is standard. For each guess in the encryption direction and for each guess in the decryption direction if we have a pair of ID characteristic, we eliminate the guesses. We already have a table for the decryption direction. As the last step of the attack, we check if all of the 2^{72} secret candidates in the encryption side lead to the impossible characteristic for each round key among 2^{160} keys (2^{144} keys for AES-192) of the decryption side in the table and delete the round key from the table. We have 128 pairs on average for each key in the table lead to the impossible characteristic in the decryption side. The corresponding plaintexts of each pair produce an impossible path with probability $1 - 2^{-8}$ for any guessed 72-bit secret value. If all the 2^{72} guesses of the secret information in the encryption side are eliminated, we delete the 160-bit (144-bit for AES-192) round key candidate from the table. The expected numbers of wrong keys left are $2^{72+160}(1-2^{-8})^{2^{15}} \approx 2^{232}e^{-128} \approx 2^{48}$ and $2^{72+144}(1-2^{-8})^{2^{15}} \approx 2^{32}$ for AES-256 and AES-192, respectively, where e is the Euler number. The remaining round keys are eliminated by exhaustive search, which costs $2^{48}2^{256-128-32} = 2^{144}$ and $2^{32}2^{192-128-64-16} = 2^{80}$ encryptions for AES-256 and AES-192, respectively.

The data complexity is only 256 chosen plaintexts. The memory complexities are 2^{168} and 2^{152} bytes for AES-256 and AES-192, respectively. It's noteworthy that storing only the first bytes of the corresponding plaintext pairs for the ciphertext pairs leading to the ID characteristic for a guessed round key in RK_6 and $MC^{-1}(RK_5)$ is sufficient. The second step dominates the time complexity. Therefore, the time complexities are $2 \cdot 2^{72} \cdot 2^{160} = 2^{233}$ and $2 \cdot 2^{72} \cdot 2^{144} = 2^{217}$ 2-round encryptions of AES-256 and AES-192, respectively. Hence, this attack has limited effectiveness.

5. Improvement and the duality of the attack

We treat our ID attack in Section 3 as an MiTM attack in this section. It is possible to collide the correct 72-bit constant guess in the encryption direction and the 160-bit subkey in the decryption direction through the MiTM technique [3, 6]. In a conventional ID attack, the wrong keys are sieved one by one by identifying the ID characteristic for the corresponding plaintext/ciphertext pairs. Unlike all the other ID attacks, we do not eliminate the wrong keys one by one.

We have constructed a table with 2^{160} and 2^{144} rows for AES-256 and AES-192 respectively in the first phase of our ID attack in Section 3. This was the process in the decryption side and we do not have any improvement in this phase. Recall that each row of our table represents a guessed subkey RK for $(RK_6, MC^{-1}(RK_5)[0, 7, 10, 13])$ and contains the first bytes of the plaintext pairs whose ciphertext pairs have a passive byte in $SR_4[0]$ when decrypting through this RK. These pairs (P_i, P_j) are enumerated and ordered by treating each pair as a number $i \cdot 2^8 + j$ within a row. Subsequently, the table is sorted in lexicographic order during its construction. Additionally, another table lists the round keys that do not decrypt any ciphertext pairs with a zero difference at SR_4. We anticipate having 2^{48} and 2^{32} such round keys for AES-256 and AES-192, respectively. This second list is used when no collision is found in the first list, indicating that the correct subkey RK is not present in the first list. Subsequently, we can search for it in the second list, which is considerably smaller.

The identification of the correct 72-bit secret value and its corresponding round key on the decryption side can be recovered within the sorted table without the need for sieving through incorrect keys. We search for the collision in the table for each 72-bit secret value. The computation of ΔSR_4 is conducted for every pair (P_i, P_j) in the encryption direction through the 72-bit guess of constant bytes, and $i \cdot 2^8 + j$ is appended to a list if $\Delta SR_4 = 0$. Upon completion of the list, a lexicographical sorting process is executed, followed by a verification step within the table of the first phase of the attack. If a match is found, the associated row number designates the correct round key for $(RK_6, MC^{-1}(RK_5)[0, 7, 10, 13])$. The time complexity associated with searching the sorted table is determined as $2^{72} \cdot 160 \approx 2^{80}$. In the absence of a match between the list and the table, signifying a 72-bit secret value where, for any pair (P_i, P_j), $\Delta SR_4 \neq 0$, the correct round key for $(RK_6, MC^{-1}(RK_5)[0, 7, 10, 13])$ is absent from the first table. Subsequently, an exhaustive search is undertaken on the second table, incurring a computational cost of $2^{48+256-160} = 2^{144}$ and $2^{32+192-144} = 2^{80}$ encryptions for AES-256 and AES-192, respectively. The attack for AES-192 is described in detail in Algorithm 1. The case for AES-256 is similar. The only difference is making search on $MC^{-1}(RK_5)[0, 13]$ rather than determining it through the key schedule.

The duality arises from the observation that any pair of plaintexts, leading to passive bytes in $\Delta SR_4[0]$ through a 72-bit constant value guess, can also form an ID characteristic if their corresponding ciphertexts result in an active byte in $\Delta MC_4^{-1}[0]$ through a 160-bit round key guess. Consequently, any 72-bit constant value guess for encryption and a 160-bit round key guess for decryption, which result in passive bytes in $\Delta SR_4[0]$, will be considered a candidate for the correct key. In summary, for the correct key pair, if the condition for the output difference of the ID characteristic in the decryption direction is satisfied for a given input/output pair, then the condition in the encryption direction is not satisfied, and vice versa. This condition does not work in two directions in general for an arbitrary ID attack. Then, the set of pairs satisfying the output difference in the decryption direction for an ID characteristic will be a subset of the set that does not satisfy the input difference for a correct guess of the subkeys. To address this, we should initially create the set for a specific subkey candidate in the encryption direction and then search for a subset of this set in the table. This task poses a greater difficulty in finding a match.

To be precise, let $(\Delta X, \Delta Y)$ be an ID characteristic. We have two lists of subkeys: one is the list of subkeys in the encryption direction, denoted as $\mathcal{L}_E$, and the other is the list of subkeys in the decryption side, denoted as $\mathcal{L}_D$. The list $\mathcal{L}_E$ contains vectors $\mathcal{V}(RK_e)$ whose i-th coordinates are 1 if the i-th plaintext pair produces the difference ΔX through encryption by RK_e, and 0 otherwise, for subkeys RK_e in the encryption direction. Similarly, the list $\mathcal{L}_D$ contains vectors $\mathcal{V}(RK_d)$ whose i-th coordinates are 1 if the i-th ciphertext pair produces the difference ΔY through decryption by RK_d, and 0 otherwise, for a subkey RK_d in the decryption direction.

Algorithm 1 MiTM attack which is dual of the ID attack in Section 3 on 6-round AES-192 with 2^8 CP

Input: Plaintext and ciphertext pairs (P_k, C_k) for $k = 0, \ldots, 2^8 - 1$

```
The table, 𝒯, for ΔSR_4[0,4], is empty for initialization
The table ℛ is empty for initialization
for each guess of RK_6 do
  Compute MC^{-1}(RK_5)[0,13]) from RK_6 using key schedule
  for each guess of MC^{-1}(RK_5)[7,10] do
    Load MC^{-1}(RK_5)[0,13]) in the t-th row of the table 𝒯 where t is the guessed subkey value
    for j from 1 to 2^8 − 1 do
      for i from 0 to j − 1 do
        Compute MC_5^{-1}[0,7,10,13] for C_i and C_j using RK_6
        Compute ΔMC_4^{-1}[0,4,8,12] using MC_5^{-1}[0,7,10,13] and MC^{-1}(RK_5)[0,7,10,13]) for C_i and C_j
        Compute SR_4[0,4] for C_i and C_j as SR_4[0,4]_i and SR_4[0,4]_j respectively
        if SR_4[0,4]_i = SR_4[0,4]_j then
          Load the value α_{i,j} = 256i + j in the t-th row of the table 𝒯 where t is the guessed subkey value
        end if
      end for
    end for
    if ∃ no α_{i,j} then
      Add RK_6 and MC^{-1}(RK_5)[0,7,10,13] to ℛ
    end if
    Sort 𝒯 with respect to α_{i,j} in lexicographic order keeping its row numbers
  end for
end for
for each guess of RK_0[0] and MC^{-1}(RK_1)[0] do
  for each guess of SB_1[5,10,15] ⊕ MC^{-1}(RK_1)[5,10,15] do
    for each guess of 𝒞_1, 𝒞_2, 𝒞_3, and 𝒞_4 do
      for each guess of RK_3[0] do
        Initialize the list ℒ as empty set
        for j from 1 to 2^8 − 1 do
          for i from 0 to j − 1 do
            Compute SR_2[0,7,10,13] for (P_i, P_j) using RK_0[0], MC^{-1}(RK_1)[0], and SB_1[5,10,15] ⊕
            MC^{-1}(RK_1)[5,10,15]
            Compute SR_3[0,4,8,12] for P_i and P_j using SR_2[0,7,10,13], 𝒞_ℓ for ℓ = 1,2,3,4
            Compute MC_3[0] using SR_3[0,4,8,12] for P_i and P_j
            Compute SR_4[0] for P_i and P_j using MC_3[0] and RK_3[0] as SR_4[0,4]_i and SR_4[0,4]_j respectively
            if SR_4[0,4]_i ≠ SR_4[0,4]_j as the duality condition then
              Load the value β_{i,j} = 256i + j in the t-th row of the table ℒ where t is the guessed subkey value
            end if
          end for
        end for
        if ℒ is equal to one of the rows of 𝒯 then
          Print the row number of 𝒯 as a candidate for the correct subkey RK_6 and MC^{-1}(RK_5)[7,10]
        else
          Make exhaustive search on ℛ for RK_6 and MC^{-1}(RK_5)[0,7,10,13]
        end if
      end for
    end for
  end for
end for
```

The miss-in-the-middle attack corresponds to finding a unique $\mathcal{V}(RK_{e_0}) \in \mathcal{L}_E$ and $\mathcal{V}(RK_{d_0}) \in \mathcal{L}_D$ such that $\mathcal{V}(RKe_0) \cdot \mathcal{V}(RK_{d_0}) = 0$, where the multiplication is a bitwise AND-operation. Therefore, we should provide enough number of data. This is a new interpretation of the generic ID attack. Then, the correct pair of subkeys will be (e_0, d_0). Recovering e_0 and d_0 by an algorithm whose complexity is less than $|\mathcal{L}_E| \cdot |\mathcal{L}_D|$ is a research problem where $|\mathcal{L}_E|$ is the dimension of the list, that is, the product of its row and column numbers. Observe that $\mathcal{V}(RK_{d_0})$ is the complementation of $\mathcal{V}(RK_{e_0})$ for the meet-in-the-middle case, and hence we have $\mathcal{V}(RK_{e_0}) \cdot \mathcal{V}(RK_{d_0}) = 0$ as a special case for the MiTM attacks. Then, it is possible to recover e_0 and d_0 in the sorted lists with a complexity of $\max\{|\mathcal{L}_E|, |\mathcal{L}_D|\}$ in this case. We also exploit that $\mathcal{V}(RK_{e_0})$ is the complementation of $\mathcal{V}(RK_{d_0})$ in our attack.

The predominant factor influencing the overall time complexity is the preparation of the ordered table. Conversely, we can enhance the efficiency of table preparation by a factor of 2^8 through the sequential decryption of 2^8 ciphertexts, as opposed to decrypting 2^{15} pairs for each round key guess. Consequently, the complexity becomes 2^{159} and 2^{143} for two-round decryptions in the cases of AES-256 and AES-192, respectively. It is important to note that there is no improvement in memory complexity.

6. A meet in the middle attack with minimum data

In this section, we present an MiTM attack using only 16 CP, where their i-th bytes are equal for $i = 1, \ldots, 15$. Similar to the attack described in Section 5, we guess 160-bit subkeys (128+32) for AES-256 and 144-bit subkeys (128+16) for AES-192 from round keys RK_6 and RK_5, respectively, on the decryption side. This is done to recover the difference in the first column of the fourth round after the SR operation, specifically $SR_4[0, 4, 8, 12]$. Our focus lies solely on the differences in two bytes, namely $SR_4[0, 4]$.

Let us remark that the pair (C_i, C_k) does not provide additional information compared to using (C_i, C_j) and (C_j, C_k). Therefore, we utilize only $16 - 1 = 15$ differences among these 16 ciphertexts, specifically the differences (C_i, C_{i+1}) for $i = 1, \ldots, 15$. We recover the difference in $SR_4[0, 4]$ for each guess and store them in a table sorted according to the differences. Each row consists of $2 \times 15 = 30$ bytes for AES-256 and $2 \times 15 + 2 = 32$ bytes for AES-192. Additional two bytes are the determined bytes of RK_5 from RK_6 through the key schedule for AES-192. Consequently, we need $20 \times 30 \times 2^{160} \approx 2^{169}$ and $18 \times 32 \times 2^{144} = 2^{153}$ bytes of memory for AES-256 and AES-192, respectively. We need to load the row numbers since we sort the table.

Firstly, we guess two subkey bytes in the encryption direction; namely $RK_0[0]$ and $MC^{-1}(RK_1)[0]$. Then, we guess three constant bytes, $SB_1[5, 10, 15] \oplus MC^{-1}(RK_1)[5, 10, 15]$, to compute $MC_1[0, 4, 8, 12]$. We further guess four constant bytes in the second round: $\mathcal{C}_1 = 3SR_2[4] \oplus SR_2[8] \oplus SR_2[12] \oplus MC^{-1}(RK_2)[0]$, $\mathcal{C}_2 = SR_2[1] \oplus 2SR_2[5] \oplus 3SR_2[9] \oplus MC^{-1}(RK_2)[5]$, $\mathcal{C}_3 = SR_2[2] \oplus SR_2[6] \oplus 3SR_2[14] \oplus MC^{-1}(RK_2)[10]$, and $\mathcal{C}_4 = 3SR_2[3] \oplus SR_2[11] \oplus 2SR_2[15] \oplus MC^{-1}(RK_2)[15]$ as defined in Section 4. Then, we can compute $MC_3[0]$. Similarly, we can guess four more constant bytes in the second round and compute $MC_3[5]$. These four constant bytes are:

$$
\begin{aligned}
\mathcal{B}_1 &= SR_2[4] \oplus SR_2[8] \oplus 2SR_2[12] \oplus MC^{-1}(RK_2)[12], \\
\mathcal{B}_2 &= 2SR_2[5] \oplus 3SR_2[5] \oplus SR_2[9] \oplus MC^{-1}(RK_2)[1], \\
\mathcal{B}_3 &= SR_2[2] \oplus 3SR_2[6] \oplus SR_2[14] \oplus MC^{-1}(RK_2)[6], \text{ and} \\
\mathcal{B}_4 &= SR_2[3] \oplus 2SR_2[11] \oplus 3SR_2[15] \oplus MC^{-1}(RK_2)[11].
\end{aligned}
$$

Once $MC_3[0,5]$ is recovered, we can compute $SR_4[0,4] = SB(MC_3[0,5] \oplus RK_3[0,5])$ by further guessing the subkey bytes $RK_3[0,5]$. In summary, the number of bits to be guessed is $16+24+32+32+16 = 120$. Then, we compute $\Delta SR_4[0,4]$ for the 15 pairs (P_i, P_{i+1}) and check if a set of these pairs is in the list for a specific 120-bit guessed value. We have less than $2^{120} \cdot \log_2(2^{165}) \approx 2^{128}$ table look-ups. The details of the attack for AES-192 is given in Algorithm 2. The approach for AES-256 is similar, with the sole difference being the guess for $MC^{-1}(RK_5)[0,13])$ rather than its derivation through the key schedule. The dominant part is $16 \cdot 2^{160} = 2^{164}$ and $16 \cdot 2^{144} = 2^{148}$ 2-round decryptions for AES-256 and AES-192, respectively. After searching the table, we expect to deduce 2^{24} and 2^{40} subkey candidates in the table for AES-256 and AES-192, respectively, since the probability that a wrong guess pair in both encryption and decryption directions produces all the 15 coinciding differences in $SR_4[0,4]$ is roughly $2^{-16 \cdot 15} = 2^{-240}$. The remaining part of the key can be recovered by the exhaustive search in much less complexity since these workloads are $2^{24} \cdot 2^{192-144} = 2^{72}$ and $2^{40} \cdot 2^{256-160} = 2^{136}$ for AES-192 and AES-256, respectively.

7. Extension of the attack on 7-round AES through integral analysis

In this section, we introduce an extension of the attack on 6-round AES in Section 6 for one more round by utilizing the key space partitioning technique introduced in [48] for integral attacks. Consider a structure of the plaintext set where $P[5,10,15]$ takes all 2^{24} values, while the rest, including $P[0]$, are held constant. We can first make a guess for $RK_0[5] \oplus RK_0[10]$ and $RK_0[5] \oplus RK_0[15]$, then select 2^8 plaintexts from the 2^{24} possibilities, such that $P[5] \oplus RK_0[5] = P[10] \oplus RK_0[10] = P[15] \oplus RK_0[15]$. This set of plaintexts is called the identically active set. Then, the inputs of the first column of the initial MC operation are in the form $[c, \alpha, \alpha, \alpha]$, where c is a constant, and α takes all 2^8 values. After the MC operation, the first column will have the form $[\beta_1, c, c, \beta_2]$ at the end of the first round, where β_1 and β_2 are permutations (see [48] for details). In other words, the second and third bytes will be constant.

We can improve the attack described in Section 6 by one more round for each guess of $RK_0[5] \oplus RK_0[10]$ and $RK_0[5] \oplus RK_0[15]$, utilizing the related 2^8 plaintexts. This time, we have two active bytes in two different columns before MC_2. Therefore, we need to guess 2 subkey bytes ($RK_1[0]$ and $RK_1[12]$) in the first round, two equivalent subkey bytes ($MC_2^{-1}(RK_2)[0,15]$), and 6-byte constants ($MC_2^{-1}[4,8,12,3,7,11]$) in the second round, along with 4-byte constants in the third round in the encryption direction to recover $MC_4[0]$ (see Section 6). These four constant bytes are:

$$\begin{aligned}
\mathcal{F}_1 &= 3SR_3[4] \oplus SR_3[8] \oplus MC^{-1}(RK_3)[0], \\
\mathcal{F}_2 &= SR_3[1] \oplus 2SR_3[5] \oplus MC^{-1}(RK_3)[5], \\
\mathcal{F}_3 &= SR_3[2] \oplus 3SR_3[14] \oplus MC^{-1}(RK_3)[10], \text{ and} \\
\mathcal{F}_4 &= SR_3[11] \oplus 2SR_3[15] \oplus MC^{-1}(RK_3)[15].
\end{aligned}$$

Moreover, if we guess $RK_4[0]$, we can compute $SR_5[0]$. In total, we are required to make guesses for 15 bytes. The details of the attack are provided in Algorithm 3 for AES-192. The attack for AES-256 follows a similar approach, differing only in the exclusion of key schedule utilization. The attack is depicted in Figure 4.

Algorithm 2 MiTM attack with constant guessing in Section 6 on 6-round AES-192 with 16 CP

Input: Plaintext and ciphertext pairs (P_k, C_k) for $k = 1, \ldots, 16$
The table, $\mathcal{T}$, for $\Delta SR_4[0,4]$, is empty for initialization
for each guess of RK_6 **do**
  Compute $MC^{-1}(RK_5)[0,13])$ from RK_6 using key schedule
  for each guess of $MC^{-1}(RK_5)[7,10]$ **do**
    Load $MC^{-1}(RK_5)[0,13])$ in the j th row of the table $\mathcal{T}$ where j is the guessed subkey value
    for each ciphertext pair (C_i, C_{i+1}) **do**
      Compute $MC_5^{-1}[0,7,10,13]$ for C_i and C_{i+1} using RK_6
      Compute $\Delta MC_4^{-1}[0,4,8,12]$ using $MC_5^{-1}[0,7,10,13]$ and $MC^{-1}(RK_5)[0,7,10,13])$
      Deduce $\Delta SR_4[0,4]$ and load it with P_i as $\Delta SR_4[0,4]_i$ in the j th row of the table $\mathcal{T}$ where j is the guessed subkey value
    end for
    Sort $\mathcal{T}$ with respect to $\Delta SR_4[0,4]_i$ in lexicographic order keeping its row numbers
  end for
end for
for each guess of $RK_0[0]$ and $MC^{-1}(RK_1)[0]$ **do**
  for each guess of $SB_1[5,10,15] \oplus MC^{-1}(RK_1)[5,10,15]$ **do**
    for each guess of $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ **do**
      for each guess of $\mathcal{B}_1$, $\mathcal{B}_2$, $\mathcal{B}_3$, and $\mathcal{B}_4$ **do**
        for each guess of $RK_3[0,5]$ **do**
          Initialize the list $\mathcal{L}$ as empty set
          for each plaintext pair (P_i, P_{i+1}) **do**
            Compute $SR_2[0,7,10,13]$ for (P_i, P_{i+1}) using $RK_0[0]$, $MC^{-1}(RK_1)[0]$, and $SB_1[5,10,15] \oplus MC^{-1}(RK_1)[5,10,15]$
            Compute $SR_3[0,4,8,12]$ and $SR_3[1,5,9,13]$ for P_i and P_{i+1} using $SR_2[0,7,10,13]$, $\mathcal{C}_\ell$, and $\mathcal{B}_\ell$ for $\ell = 1,2,3,4$
            Compute $MC_3[0]$ using $SR_3[0,4,8,12]$ for P_i and P_{i+1}
            Compute $MC_3[5]$ using $SR_3[1,5,9,13]$ for P_i and P_{i+1}
            Compute $SR_4[0,4]$ for P_i and P_{i+1} using $MC_3[0,5]$ and $RK_3[0,5]$
            Compute $\Delta SR_4[0,4]$ using $SR_4[0,4]$ for P_i and P_{i+1}; and add it as $\Delta SR_4[0,4]_i$ to the list $\mathcal{L}$
          end for
          if $\mathcal{L}$ is equal to one of the rows of $\mathcal{T}$ **then**
            Print the row number of $\mathcal{T}$ as a candidate for the correct subkey RK_6 and $MC^{-1}(RK_5)[7,10]$
          end if
        end for
      end for
    end for
  end for
end for

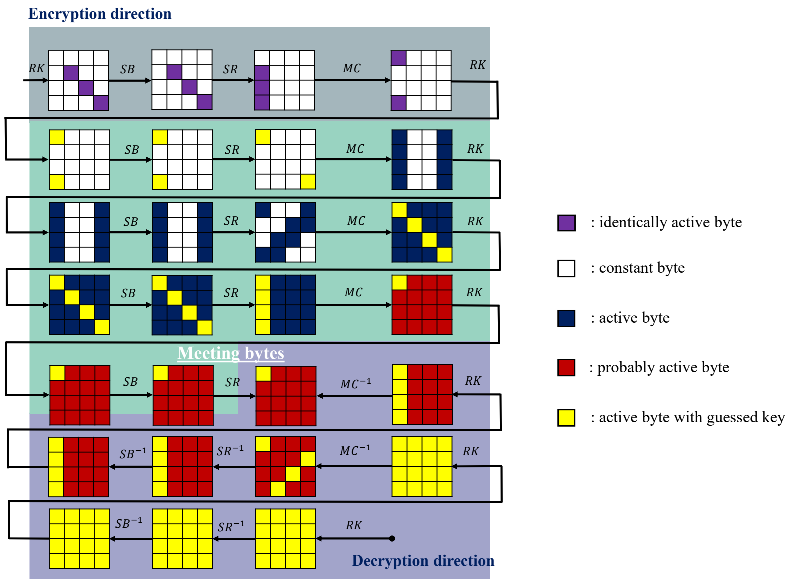

Figure 4. 7-round meet in the middle attack.

The process in the decryption side is almost the same as in Section 6 for each guess of $RK_0[5] \oplus RK_0[10]$ and $RK_0[5] \oplus RK_0[15]$. It is enough to use 63 differences since the probability that all the 63 differences coincides from encryption and decryption directions is $2^{-8 \cdot 63} = 2^{-504}$. Thus, the time complexity is $64 \cdot 2^{160} = 2^{166}$ and $64 \cdot 2^{144} = 2^{150}$ 2-round decryptions for AES-256 and AES-192, respectively, for each guess in the whitening key. We compute the differences for 2^{120} secret parameters in the encryption direction and then 2^{120} table look-ups. Thus, the complexity in the decryption side is dominant and repeated 2^{16} times since we have 2^{16} guesses for $RK_0[5] \oplus RK_0[10]$ and $RK_0[5] \oplus RK_0[15]$. Therefore, the time complexities are 2^{182} and 2^{166} two-round decryptions for AES-256 and AES-192, respectively. The data complexity is 2^{24} CP, which is the minimum among all the attacks on 7-round AES. We use 63 differences instead of 15 as in the previous section, with each difference representing one byte instead of two. Specifically, the memory complexity is twice that of the attack in Section 6, as the data is doubled, and we can reuse the memory for each guess of $RK_0[5] \oplus RK_0[10]$ and $RK_0[5] \oplus RK_0[15]$.

We can improve the data complexity further. Consider one structure of the plaintext set where $P[10, 15]$ takes all the 2^{16} values and the remaining bytes are all kept constant. We can first make a guess for $RK_0[10] \oplus RK_0[15]$ and select 2^8 plaintexts among 2^{16} of them such that $P[10] \oplus RK_0[10] = P[15] \oplus RK_0[15]$. Then, the inputs of the first column of the first MC operation are of the form $[c, c, \alpha, \alpha]$ where c stands for a constant and α takes all the 2^8 values. After the MC operation, the first column will be of the form $[c, \beta_1, \beta_2, \beta_3]$ where β_is are also permutations [48]. That is, the first byte will be constant for all 2^8 plaintexts. Therefore, we must guess 3 subkey bytes in the second round; 3 equivalent subkey bytes and 9-byte constants ($MC_2^{-1}[1, 9, 13, 2, 6, 14, 3, 11, 15]$) in the third round; and 4-byte constants in the fourth round in the encryption direction to recover $MC_4[0]$. These four constants are $2SR_3[0] \oplus MC^{-1}(RK_3)[0]$, $SR_3[13] \oplus MC^{-1}(RK_3)[5]$,

Algorithm 3 Integral-MiTM attack with constant guessing in Section 7 on 7-round AES-192 with 2^{24} CP

Input: Plaintext and ciphertext pairs (P_k, C_k) for $k = 1, \ldots, 2^{24}$
for each guess of $RK_0[5] \oplus RK_0[10]$ and $RK_0[5] \oplus RK_0[15]$ **do**
  Select 64 plaintexts P_i satisfying $P_i[5] \oplus P_i[10] = RK_0[5] \oplus RK_0[10]$ and $P_i[5] \oplus P_i[15] = RK_0[5] \oplus RK_0[15]$ for $i = 1, \ldots, 64$
  The table, $\mathcal{T}$, for $\Delta SR_5[0,4]$, is empty for initialization
  for each guess of RK_7 **do**
    Compute $MC^{-1}(RK_6[0,13])$ from RK_7 using key schedule
    for each guess of $MC^{-1}(RK_6[7,10]$ **do**
      Load $MC^{-1}(RK_6[0,13])$ in the j th row of the table $\mathcal{T}$ where j is the guessed subkey value
      for each ciphertext pair (C_i, C_{i+1}) for $i = 1, \ldots, 63$ **do**
        Compute $MC_6^{-1}[0,7,10,13]$ for C_i and C_{i+1} using RK_7
        Compute $\Delta MC_5^{-1}[0,4,8,12]$ using $MC_6^{-1}[0,7,10,13]$ and $MC^{-1}(RK_6[0,7,10,13])$
        Deduce $\Delta SR_5[0,4]$ and load it with P_i as $\Delta SR_5[0,4]_i$ in the j th row of the table $\mathcal{T}$ where j is the guessed subkey value
      end for
      Sort $\mathcal{T}$ with respect to $\Delta SR_5[0,4]_i$ in lexicographic order keeping its row numbers
    end for
  end for
  for each subkey guess of $RK_1[0]$, $RK_1[12]$, and $MC_2^{-1}(RK_2)[0,15]$ **do**
    for each constant guess of $MC_2^{-1}[4,8,12,3,7,11]$ **do**
      for each constant guess of $\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_3$, and $\mathcal{F}_4$ **do**
        for each guess of $RK_4[0]$ **do**
          Initialize the list $\mathcal{L}$ as empty set
          for each plaintext pair (P_i, P_{i+1}) **do**
            Compute $SB_3[0,4,8,12]$ using $RK_1[0]$ and $MC_2^{-1}[4,8,12]$
            Compute $SB_3[3,7,11,15]$ using $RK_1[12]$ and $MC_2^{-1}[3,7,11]$
            Compute $SB_4[0,5,10,15]$ using $SB_3[0,4,8,12]$, $SB_3[3,7,11,15]$, and $\mathcal{F}_\ell$ for $\ell = 1,2,3,4$.
            Compute $MC_4[0]$ using $SB_4[0,5,10,15]$
            Compute $SR_5[0]$ using $MC_4[0]$ and $RK_4[0]$
            Deduce the difference $\Delta SR_5[0]_i$ for P_i and P_{i+1}
            Add $\Delta SR_4[0,4]_i$ to the list $\mathcal{L}$
          end for
          if $\mathcal{L}$ is equal to one of the rows of $\mathcal{T}$ **then**
            Print the row number of $\mathcal{T}$ as a candidate for the correct subkey RK_7 and $MC^{-1}(RK_6[7,10])$
          end if
        end for
      end for
    end for
  end for
end for

$2SR_3[10] \oplus MC^{-1}(RK_3)[10]$, and $SR_3[7] \oplus MC^{-1}(RK_3)[15]$. In summary, adding $RK_4[0]$ in our guesses, we need to guess $152 + 8 = 160$ bits to recover $SR_5[0]$. This time, it is enough to use 63 differences and the time complexity of preparing the table in the decryption side is $64 \cdot 2^{160} = 2^{166}$ 2-round decryptions for AES-256 for each guess of $RK_0[10] \oplus RK_0[15]$. We compute the differences for 2^{160} secret parameters in the encryption direction and then 2^{160} table look-ups. Hence, the dominant part is $2^{166+8} = 2^{174}$ 2-round decryptions for AES-256, which is slightly below the cost of 2^{173} encryptions. On the other hand, the dominant part of the time complexity for AES-192 is 2^{168} table look-ups which consists of approximately 2^{175} vector comparisons. This is equivalent to around 2^{171} encryptions. The data complexity is only 2^{16} CP. The memory complexity does not change.

8. Conclusion

We have studied the low data attacks on 6-round and 7-round AES and achieved new records. We have shown that only 16 CP is enough to recover the key faster than the exhaustive search for 6-round AES-192 and AES-256. We also have mounted an attack using 2^{16} CP on 7-round AES-256 and AES-192. We have achieved these low data complexities by utilizing constant guessing techniques and combining the MiTM attacks with square and integral attacks. The constant guessing technique can be utilized to improve the attacks based on pairs of inputs through passive words such as ID attacks, differential attacks, truncated differential attacks, boomerang attacks etc. When these attacks are mounted on word-oriented square type algorithms, it is possible to extend them to further rounds through the constant guessing technique to achieve the best complexity records.

A new SPN construction technique with a new linear transformation layer method providing ”second degree diffusion” is embodied on a new cipher called DIZY [50]. The SPNs having such diffusion layers are not word-oriented and hence the constant guessing technique will probably not work on such ciphers.

We think that our attacks do not work for the full round and hence do not threaten the security of AES. Moreover, mounting a square-ID attack on 6-round AES-128 minimizing data is left as open question. The perfectly fast diffusion property of the linear layer of AES, consisting of the SR and the MC operations, hinders the extension of our attacks to 8 or more rounds with the existing data amounts. As one corollary of this fast diffusion, we conjecture that there is no attack faster than the brute force on AES-128 with more than 5 rounds if there are only 16 blocks of plaintexts or ciphertexts available.

Acknowledgment

We are grateful to the anonymous reviewers and the associated editor for their invaluable comments. The author is partially supported by TÜBİTAK 1001 Project under the grant number 121E228.

References

[1] Dworkin M, Barker E, Nechvatal J, Foti J, Bassham L et al. Advanced encryption standard (AES), 2001-11-26 2001. https://doi.org/10.6028/NIST.FIPS.197

[2] Demirci H, Selçuk AA. A meet-in-the-middle attack on 8-round AES. In Kaisa Nyberg, editor, Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers, volume 5086 of Lecture Notes in Computer Science, pages 116–126. Springer, 2008. https://doi.org/10.1007/978-3-540-71039-4_7

[3] Dunkelman O, Keller N, Shamir A. Improved single-key attacks on 8-round AES-192 and AES-256. Journal of Cryptology 2015;28 (3):397–422 https://doi.org/10.1007/s00145-013-9159-4

[4] Dunkelman O, Keller N, Shamir A. Improved single-key attacks on 8-round AES-192 and AES-256. In Masayuki Abe, editor, Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings, volume 6477 of Lecture Notes in Computer Science, pages 158–176. Springer, 2010. https://doi.org/10.1007/978-3-642-17373-8_10

[5] Wang G, Zhu C. Single key recovery attacks on reduced AES-192 and Kalyna-128/256. Science China Information Sciences 2017; 60 (9):99101 https://doi.org/10.1007/s11432-016-0417-7

[6] Derbez P, Fouque P, Jean J. Improved key recovery attacks on reduced-round AES in the single-key setting. In Thomas Johansson and Phong Q. Nguyen, editors, Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings, volume 7881 of Lecture Notes in Computer Science, pages 371–387. Springer, 2013. https://doi.org/10.1007/978-3-642-38348-9_23

[7] Li L, Jia K, Wang X. Improved single-key attacks on 9-round AES-192/256. In Carlos Cid and Christian Rechberger, editors, Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers, volume 8540 of Lecture Notes in Computer Science, pages 127–146. Springer, 2014. https://doi.org/10.1007/978-3-662-46706-0_7

[8] Gilbert H, Minier M. A collision attack on 7 rounds of Rijndael. In The Third Advanced Encryption Standard Candidate Conference, April 13-14, 2000, New York, New York, USA, pages 230–241. National Institute of Standards and Technology, 2000.

[9] Ferguson N, Kelsey J, Lucks S, Schneier B, Stay M et al. Improved cryptanalysis of Rijndael. In Bruce Schneier, editor, Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings, volume 1978 of Lecture Notes in Computer Science, pages 213–230. Springer, 2000. https://doi.org/10.1007/3-540-44706-7_15.

[10] Bogdanov A, Khovratovich D, Rechberger C. Biclique cryptanalysis of the full AES. In Dong Hoon Lee and Xiaoyun Wang, editors, Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings, volume 7073 of Lecture Notes in Computer Science, pages 344–371. Springer, 2011. https://doi.org/10.1007/978-3-642-25385-0_19

[11] Tao B, Wu H. Improving the biclique cryptanalysis of AES. In Ernest Foo and Douglas Stebila, editors, Information Security and Privacy - 20th Australasian Conference, ACISP 2015, Brisbane, QLD, Australia, June 29 - July 1, 2015, Proceedings, volume 9144 of Lecture Notes in Computer Science, pages 39–56. Springer, 2015. https://doi.org/10.1007/978-3-319-19962-7_3

[12] Saha D, Rahman M, Paul G. New yoyo tricks with AES-based permutations. IACR Transactions on Symmetric Cryptology 2018;(4):102–127 2018. https://doi.org/10.13154/tosc.v2018.i4.102-127

[13] Rahman M, Saha D, Paul G. Boomeyong: Embedding yoyo within boomerang and its applications to key recovery attacks on AES and pholkos. IACR Transactions on Symmetric Cryptology 2021;(3):137–169 2021. https://doi.org/10.46586/tosc.v2021.i3.137-169

[14] Bariant A, Leurent G. Truncated boomerang attacks and application to AES-based ciphers. In Carmit Hazay and Martijn Stam, editors, Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV, volume 14007 of Lecture Notes in Computer Science, pages 3–35. Springer, 2023. https://doi.org/10.1007/978-3-031-30634-1_1

[15] Bardeh N, Rijmen V. New key-recovery attack on reduced-round AES. IACR Transactions on Symmetric Cryptology 2022 (2):43–62 2022. https://doi.org/10.46586/tosc.v2022.i2.43-62

[16] Zhao K, Cui J, Xie Z. Algebraic cryptanalysis scheme of AES-256 using Gröbner basis. Journal of Electrical and Computer Engineering, 2017:9828967:1–9828967:9, 2017. https://doi.org/10.1155/2017/9828967

[17] Grassi L. Probabilistic mixture differential cryptanalysis on round-reduced AES. In Kenneth G. Paterson and Douglas Stebila, editors, Selected Areas in Cryptography - SAC 2019 - 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019, Revised Selected Papers, volume 11959 of Lecture Notes in Computer Science, pages 53–84. Springer, 2019. https://doi.org/10.1007/978-3-030-38471-5_3

[18] Grassi L, Schofnegger M. Mixture integral attacks on reduced-round AES with a known/secret s-box. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings, volume 12578 of Lecture Notes in Computer Science, pages 312–331. Springer, 2020. https://doi.org/10.1007/978-3-030-65277-7_14

[19] Boura C, Lallemand V, Naya-Plasencia M, Suder V. Making the impossible possible. Journal of Cryptology 2018;31 (1):101–133 https://doi.org/10.1007/s00145-016-9251-7

[20] He Z, Hu Z. A new method for impossible differential cryptanalysis of 8-round AES-128. In 2nd International Symposium on Intelligence Information Processing and Trusted Computing, IPTC 2011, Wuhan, China, October 22-23, 2011, pages 214–217. IEEE Computer Society, 2011. https://doi.org/10.1109/IPTC.2011.62

[21] Hu Z, He Z. A new method for impossible differential cryptanalysis of 7-round AES-192. In 2nd International Symposium on Intelligence Information Processing and Trusted Computing, IPTC 2011, Wuhan, China, October 22-23, 2011, pages 248–251. IEEE Computer Society, 2011. https://doi.org/10.1109/IPTC.2011.73

[22] Jiang Z, Jin C, Wang Z. Multiple impossible differentials attack on AES-192. IEEE Access, 7:138011–138017, 2019. https://doi.org/10.1109/ACCESS.2019.2942960

[23] Lu J, Dunkelman O, Keller N, Kim J. New impossible differential attacks on AES. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, 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, pages 279–293. Springer, 2008. https://doi.org/10.1007/978-3-540-89754-5_22

[24] Luo Y, Lai X. Improvements for finding impossible differentials of block cipher structures. Security and Communication Networks, 2017:5980251:1–5980251:9, 2017. https://doi.org/10.1155/2017/5980251

[25] Liu Y, Shi Y, Gu D, Dai B, Zhao F et al. Improved impossible differential cryptanalysis of large-block Rijndael. Science China Information Sciences 2019;62 (3):32101:1–32101:14 https://doi.org/10.1007/s11432-017-9365-4

[26] Mala H, Dakhilalian M, Rijmen V, Modarres-Hashemi M. Improved impossible differential cryptanalysis of 7-round AES-128. In Guang Gong and Kishan Chand Gupta, editors, Progress in Cryptology - INDOCRYPT 2010 - 11th International Conference on Cryptology in India, Hyderabad, India, December 12-15, 2010. Proceedings, volume 6498 of Lecture Notes in Computer Science, pages 282–291. Springer, 2010. https://doi.org/10.1007/978-3-642-17401-8_20

[27] Phan RCW. Impossible differential cryptanalysis of 7-round Advanced Encryption Standard (AES). Information Processing Letters 2004;91 (1):33–38. https://doi.org/10.1016/j.ipl.2004.02.018

[28] Zhang W, Wu W, Feng D. New results on impossible differential cryptanalysis of reduced AES. In Kil-Hyun Nam and Gwangsoo Rhee, editors, Information Security and Cryptology - ICISC 2007, 10th International Conference, Seoul, Korea, November 29-30, 2007, Proceedings, volume 4817 of Lecture Notes in Computer Science, pages 239–250. Springer, 2007. https://doi.org/10.1007/978-3-540-76788-6_19

[29] Zhang M, Zhang W, Liu J, Wang X. General impossible differential attack on 7-round AES. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 2010; 93-A(1):327–330 https://doi.org/10.1587/transfun.E93.A.327

[30] Grassi L, Rechberger C, Rønjom S. A new structural-differential property of 5-round AES. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part II, volume 10211 of Lecture Notes in Computer Science, pages 289–317, 2017. https://doi.org/10.1007/978-3-319-56614-6_10

[31] Bahrak B, Aref MR. Impossible differential attack on seven-round AES-128. IET Information Security 2008;2 (2):28–32. https://doi.org/10.1049/iet-ifs:20070078

[32] Cheon J, Kim M, Kim K, Lee JY, Kang SW. Improved impossible differential cryptanalysis of Rijndael and Crypton. In Kwangjo Kim, editor, Information Security and Cryptology - ICISC 2001, 4th International Conference Seoul, Korea, December 6-7, 2001, Proceedings, volume 2288 of Lecture Notes in Computer Science, pages 39–49. Springer, 2001. https://doi.org/10.1007/3-540-45861-1_4

[33] Leurent G, Pernot C. New representations of the AES key schedule. In Anne Canteaut and François-Xavier Standaert, editors, Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part I, volume 12696 of Lecture Notes in Computer Science, pages 54–84. Springer, 2021. https://doi.org/10.1007/978-3-030-77870-5_3

[34] Knudsen L. DEAL a 128-bit block cipher. Complexity 1998;258 (2):216.

[35] Biham E, Biryukov A, Shamir A. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In International Conference on the Theory and Applications of Cryptographic Techniques, pages 12–23. Springer, 1999.

[36] Biham E, Shamir A. Differential cryptanalysis of the full 16-round DES. In Ernest F. Brickell, editor, Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings, volume 740 of Lecture Notes in Computer Science, pages 487–496. Springer, 1992. https://doi.org/10.1007/3-540-48071-4_34.

[37] Matsui M. Linear cryptanalysis method for DES cipher. In Tor Helleseth, editor, Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings, volume 765 of Lecture Notes in Computer Science, pages 386–397. Springer, 1993. https://doi.org/10.1007/3-540-48285-7_33

[38] Bar-On A, Dunkelman O, Keller N, Ronen E, Shamir A. Improved key recovery attacks on reduced-round AES with practical data and memory complexities. Journal of Cryptology 2020;33 (3):1003–1043. https://doi.org/10.1007/s00145-019-09336-w

[39] Bouillaguet C, Derbez P, Dunkelman O, Fouque P, Keller N et al. Low-data complexity attacks on AES. IEEE Transactions on Information Theory 2012;58 (11):7002–7017. https://doi.org/10.1109/TIT.2012.2207880

[40] Bar-On A, Dunkelman O, Keller N, Ronen E, Shamir A. Improved key recovery attacks on reduced-round AES with practical data and memory complexities. In Hovav Shacham and Alexandra Boldyreva, editors, Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II, volume 10992 of Lecture Notes in Computer Science, pages 185–212. Springer, 2018. https://doi.org/10.1007/978-3-319-96881-0_7

[41] Daemen J, Rijmen V. The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography. Springer, 2002. https://doi.org/10.1007/978-3-662-04722-4

[42] Bardeh N, Rønjom S. Practical attacks on reduced-round AES. In Johannes Buchmann, Abderrahmane Nitaj, and Tajje-eddine Rachidi, editors, Progress in Cryptology - AFRICACRYPT 2019 - 11th International Conference on Cryptology in Africa, Rabat, Morocco, July 9-11, 2019, Proceedings, volume 11627 of Lecture Notes in Computer Science, pages 297–310. Springer, 2019. https://doi.org/10.1007/978-3-030-23696-0_15

[43] Pal D, Ali M, Das A, Chowdhury DR. A cluster-based practical key recovery attack on reduced-round AES using impossible-differential cryptanalysis. Journal of Supercomputing 2023;79 (6):6252–6289 https://doi.org/10.1007/s11227-022-04872-y

[44] Kara O. New security proofs and complexity records for Advanced Encryption Standard. IEEE Access, 11:131205–131220, 2023. https://doi.org/10.1109/ACCESS.2023.3335271

[45] Derbez P. Meet-in-the-Middle Attacks on AES. Theses, Ecole Normale Supérieure de Paris - ENS Paris, December 2013. https://theses.hal.science/tel-00918146.

[46] Daemen J, Knudsen LR, Rijmen V. The block cipher Square. In Eli Biham, editor, Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings, volume 1267 of Lecture Notes in Computer Science, pages 149–165. Springer, 1997. https://doi.org/10.1007/BFb0052343

[47] Dunkelman O, Ghosh S, Keller N, Leurent G, Marmor A et al. Partial sums meet FFT: improved attack on 6-round AES. IACR Cryptol. ePrint Arch., page 1659, 2023. https://eprint.iacr.org/2023/1659.

[48] Demirbaş F, Kara O. Integral characteristics by keyspace partitioning. Designs, Codes and Cryptography 2022;90 (2):443–472 https://doi.org/10.1007/s10623-021-00989-y

[49] Grassi L, Rechberger C, Rønjom S. Subspace trail cryptanalysis and its applications to AES. IACR Transactions on Symmetric Cryptology 2016;(2):192–225 Feb. 2017. https://doi.org/10.13154/tosc.v2016.i2.192-225

[50] Gül Ç, Kara O. A new construction method for keystream generators. IEEE Trans. Inf. Forensics Secur., 18:3735–3744, 2023. https://doi.org/10.1109/TIFS.2023.3287412