The Effects of the Omission of Last Round's MixColumns on AES*

Orr Dunkelman and Nathan Keller**

Faculty of Mathematics and Computer Science
Weizmann Institute of Science
P.O. Box 26, Rehovot 76100, Israel
{orr.dunkelman,nathan.keller}@weizmann.ac.il

Abstract. The Advanced Encryption Standard (AES) is the most widely deployed block cipher. It follows the modern iterated block cipher approach, iterating a simple round function multiple times. The last round of AES slightly differs from the others, as a linear mixing operation (called MixColumns) is omitted from it.
Following a statement of the designers, it is widely believed that the omission of the last round MixColumns has no security implications. As a result, the majority of attacks on reduced-round variants of AES assume that the last round of the reduced-round version is free of the MixColumns operation.
In this note we refute this belief, showing that the omission of MixColumns does affect the security of (reduced-round) AES. First, we consider a simple example of 1-round AES, where we show that the omission reduces the time complexity of an attack with a single known plaintext from 2^{48} to 2^{16}. Then, we examine several previously known attacks on 7-round AES-192 and show that the omission reduces their time complexities by a factor of 2^{16}.

1 Introduction

Since its selection in 2001, the Advanced Encryption Standard [12] has became the world's most popular block cipher. The public assessment process that was held by NIST (which ended in selecting Rijndael as the AES), the security assurances, and the performance edge have all contributed to its quick acceptance and deployment.

AES is an iterated block cipher which supports 128-bit blocks and keys of 128, 192, or 256 bits. For each key length, the round function is iterated a different number of times (10,12, and 14, respectively). A 128-bit plaintext is treated as a byte matrix of size 4x4, where each byte represents a value in $GF(2^8)$. Then, a round function composed of the following four operations is applied (as many rounds as needed):

* Some of the research described in this paper was discussed during the Early Symmetric Crypto (ESC) seminar, Remich, Luxembourg.

** The second author was partially supported by the Koshland center for basic research.

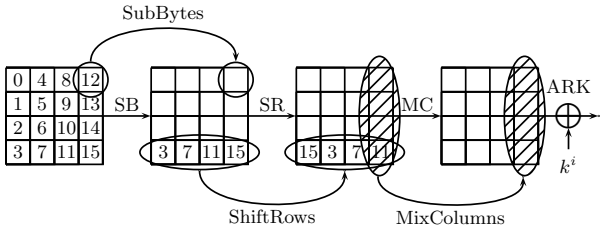

Fig. 1. An AES round

- SubBytes (SB) — a nonlinear byte-wise substitution that applies the same 8×8 S-box to every byte.
- ShiftRows (SR) — a cyclic shift of the i'th row by i bytes to the left.
- MixColumns (MC) — a matrix multiplication over a finite field applied to each column.
- AddRoundKey (ARK) — an exclusive-or with the round subkey k^r.

We outline a full AES round in Figure 1.

To ensure symmetry between encryption and decryption, two changes were made: Before the first round, an additional AddRoundKey operation is performed, and in the last round, the MixColumns operation is omitted. This omission was described by the AES designers [3] as follows:

> "In order to make the cipher and its inverse more similar in structure, the linear mixing layer of the last round is different from the mixing layer in the other rounds. It can be shown that this does not improve or reduce the security of the cipher in any way. This is similar to the absence of the swap operation in the last round of the DES."

Following this statement, it was widely believed that the omission of MixColumns indeed has no effect on the security. As a result, in most of the attacks on reduced-round variants of AES, the last round is assumed to lack MixColumns (see, for example, [1, 10, 11]).

The omission was conjectured to offer no security loss, as the AddRoundKey and the MixColumns operations in the last round can be interchanged, both being linear transformations over $GF(2^8)$ (see [4, Chapter 3.7.2]). The only "price" of this change is altering the subkey which is XORed in the AddRoundKey operation, from k^r to $MC^{-1}(k^r)$ (which is the application of the inverse $MC(\cdot)$ to each column of k^r independently). After this change, the MixColumns indeed becomes cryptographically insignificant, as it is an un-keyed operation applied to the known ciphertext.

However, we show in this letter that the omission of MixColumns is not innocent, since the altering of the last round key affects the security with respect to attacks which exploit relations between the subkeys. Indeed, the key schedule of AES is relatively simple, and the knowledge of two (specific) bytes in a round

subkey allows an adversary to deduce the value of another byte in the previous round subkey. Such deduction is problematic when the last round key k^r is replaced by $MC^{-1}(k^r)$, since then the basic relations between the two last round subkeys involve at least six bytes. As a result, the time complexity of attacks based on guessing subkey material in the last two rounds may increase when the last MixColumns exists.

We first demonstrate this observation by a simple example — an attack on 1-round AES-128 (i.e., AES with a 128-bit key) with a single known plaintext/ciphertext pair. Since the adversary has only 128 bits of information, any attack on this variant must use the relation between the two subkeys used in the encryption process. If the MixColumns operation exists, then any relation between these two subkeys requires examination of a full column, and thus it is unlikely to find an attack on this variant requiring less than 2^{32} operations. Moreover, the best reported attack on this variant [5] requires 2^{48} 1-round encryptions. On the other hand, when MixColumns is omitted, we present an attack requiring only 2^{16} 1-round encryptions.

We then examine several known attacks on 7-round AES-192 (i.e., AES with a 192-bit key). We show that the omission of MixColumns in the last round reduces the time complexities of the attacks presented by Zhang et al. [13] and by Lu et al. [9] by a multiplicative factor of 2^{16}.

We conclude this introduction with a brief description of the key schedule of AES and the notations used in the next sections.

1.1 Key Schedule and Notation

The AES key schedule initializes an array of subkey material of 32-bit words $W[\cdot]$, whose size is $4(Nr+1)$, where Nr is the number of rounds, and Nk is the number of 32-bit words in the key (e.g., for AES-128, $Nr = 10, Nk = 4$):

- For $i = 0, \ldots, Nk$: $W[i] \leftarrow K[i]$, where $K[\cdot]$ is the user supplied key.
- For $i = Nk, \ldots, 4(Nr+1)$:
  1. $temp \leftarrow W[i-1]$.
  2. If $i \bmod Nk \equiv 0$: $temp \leftarrow SB(RotWord(temp)) \oplus RCON[i/Nk]$.
  3. If $Nk = 8$ and $i \bmod 8 \equiv 4$: $temp \leftarrow SB(temp)$.
  4. $W[i] \leftarrow W[i-Nk] \oplus temp$,

where $RCON[\cdot]$ is an array of constants, and $RotWord(\cdot)$ takes 4 bytes and rotates them by one byte position to the left.

The round numbering is $0, 1, \ldots$, and the subkey used in the AddRoundKey operation in the end of round r is denoted by k^r. The initial whitening key is k^{-1}. Each subkey is represented as a byte matrix of size 4x4 (corresponding to the state matrix), and the j'th byte in the i'th row of the matrix is denoted by $k^r_{i,j}$ (for $0 \le i, j \le 3$). The "equivalent" key obtained when the MixColumns and AddRoundKey operations are interchanged is denoted by $w^r = MC^{-1}(k^r)$.

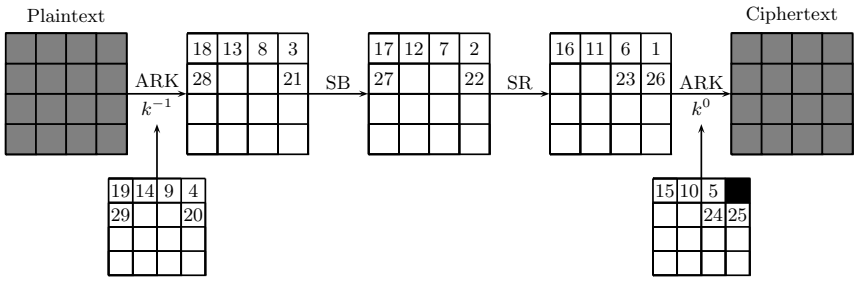

Fig. 2. The First Step of the Attack

2 A Simple Example — 1-round AES-128

We first explore the very simple case of reduced-round AES-128 with only one round, where the adversary has access to a single known plaintext/ciphertext pair. While the best reported attack on this variant including the MixColumns requires 2^{48} one-round encryptions [5], we present an attack that requires only 2^{16} encryptions, given that MixColumns is absent.

Our attack is composed of four steps, which are outlined in Figures 2–5. In each figure, the gray color denotes bytes that are already known before the respective step, and the black color denotes the byte that is guessed in the beginning of the step. The numbers of the bytes show the order in which the bytes are derived, and the bytes marked by circle are computed in two different ways and used for consistency check (i.e., if the computed values do not match, then the key guess is discarded).

In the first step of the attack, outlined in Figure 2, the adversary starts with guessing byte $k^0_{0,3}$, and in total she succeeds to retrieve six bytes of each of the subkeys k^{-1} and k^0. The bytes denoted by $5, 10, 15, 20$, and 25 are derived using the key schedule, and the rest of the bytes are derived using partial encryption/decryption.

In the second step of the attack, performed for each of the 2^8 possible values from the first step, the adversary guesses the value of $k^0_{1,0}$. The bytes denoted by 5 and 10 are derived using the key schedule, and the rest of the bytes are derived using partial encryption/decryption. The only exception is the byte denoted by 9 which is derived in two different ways in parallel (both using the key schedule and using encryption/decryption). Then the computed values are used as a consistency check: If they disagree (which occurs with probability of about $1 - 2^{-8}$) then the whole guess of $(k^0_{0,3}, k^0_{1,0})$ can be discarded. As a result, on average only 2^8 possible candidates remain after this step. Step 2 is outlined in Figure 3.

In the third step of the attack, outlined in Figure 4, for each of the 2^8 possible candidates, the adversary guesses the value of $k^0_{2,0}$. The bytes denoted by $5, 6$, and 11 are derived using the key schedule, and the byte denoted by circle is derived both using the key schedule and using partial encryption/decryption,

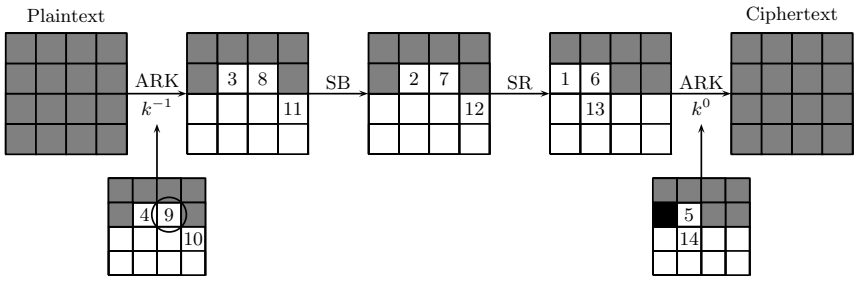

Fig. 3. The Second Step of the Attack

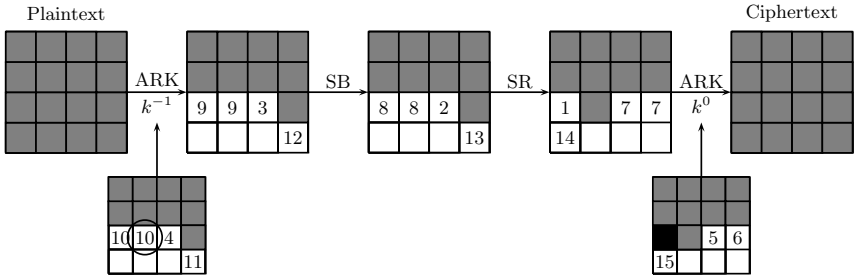

Fig. 4. The Third Step of the Attack

and used for consistency check. The rest of the bytes are derived using partial encryption/decryption. Due to the consistency check, about 2^8 candidates are expected to remain after this step.

Finally, in the fourth step of the attack, for each of the 2^8 remaining candidates, the adversary guesses byte $k^0_{3,3}$. The byte denoted by 1 is derived using the key schedule. The byte denoted by 6 is derived in two independent ways, both using the key schedule, and used for consistency check. The other byte denoted by circle is obtained using the key schedule and compared with its previously known value as a consistency check. The rest of the bytes are derived using partial encryption/decryption. Since two consistency checks are performed in this step, it is expected that only one key suggestion remains. We note that it is possible that more than one candidate remains, and then the adversary will not be able to decide which candidate is the correct key. Indeed, it is possible that the encryption of a single plaintext under two different keys yields the same ciphertext, and in this case, the information available to the adversary is not sufficient to retrieve the key uniquely.

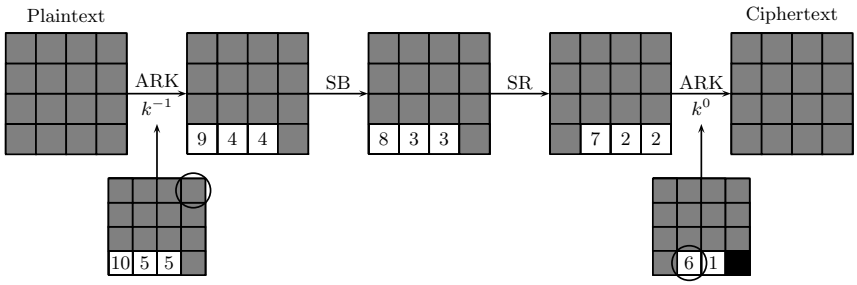

Fig. 5. The Fourth Step of the Attack

The time complexity of the above attack procedure is about 2^{16} trial encryptions (the analysis steps themselves are very lightweight and take less the a full round encryption).

3 Impossible Differential Attacks on 7-round AES-192

The basic impossible differential attack on 7-round AES-192 presented by Phan [11], was later improved in [9, 13] using several techniques and observations. The main improvement was a reduction of the time complexity by a factor of 2^{24} using subkey relations. The observation behind this improvement is the following: In Phan's attack, the adversary has to guess eight bytes of the subkey k^6 and eight bytes of the subkey k^5 in order to check whether the impossible differential holds (see in Figure 6). However, it appears that it is sufficient to guess 13 out of these 16 bytes, and the other three bytes can be deduced using the key schedule. Specifically, we have the following relations:

1. $k^6_{2,3} = k^6_{2,2} \oplus k^5_{1,2}$,
2. $k^6_{2,2} = k^5_{0,2} \oplus SB(k^6_{1,3}) \oplus RCON[5]$,
3. $k^6_{2,3} = k^5_{0,3} \oplus SB(k^6_{1,0}) \oplus RCON[5]$.

These relations, observed independently in [13] and in [9], reduce the time complexity of Phan's attack by a factor of 2^{24}. However, this reduction holds only under the assumption that the MixColumns operation in the last round is omitted. If the MixColumns exists, then instead of the subkey bytes in k^6, the adversary must guess the eight corresponding bytes of the equivalent subkey w^6 (otherwise, she has to guess the entire k^6 due to the MixColumns operation). Since passing from w^6 to k^6 requires the knowledge of a complete column, neither of the relations above can be used.

This problem can be partially solved by replacing the second-to-last subkey k^5 by the equivalent subkey w^5. We observe that since the MixColumns operation is linear, relation (1) above is equivalent to:

$$w^6_{2,3} = w^6_{2,2} \oplus w^5_{1,2}. \tag{1}$$

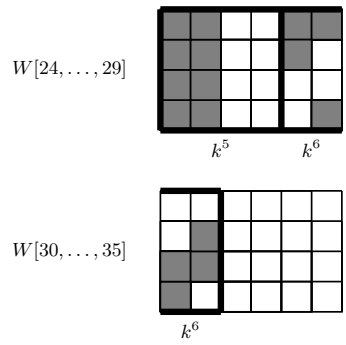

Fig. 6. The Subkey Bytes Guessed in Phan's Attack

Since the adversary guesses two full columns in k^5, she can retrieve the two corresponding columns in w^5, and then the above relation can be used.

However, this still does not help with relations (2) and (3). Due to the non-linearity of the SubBytes operation, the terms $SB(k^6_{1,3})$ and $SB(k^6_{1,0})$ cannot be replaced by terms depending on w^6, and thus these two relations cannot be used. Therefore, the omission of the last round MixColumns reduces the complexity of the attacks in [9, 13] by a factor of 2^{16}.

4 Discussion

It seems that unlike the examples presented above, there are some classes of attacks that are not affected by the omission of MixColumns. Obviously, attacks which do not exploit relations between the last round subkey and other subkeys, such as the attack on 6-round AES presented in [7] and the recent related-key attack on the full AES-192 and AES-256 presented in [2], are immune to this issue.

Other attacks, such as the 7-round attack presented in [7] and the 8-round attack presented in [6], are not affected since the adversary guesses the full last round subkey. This guess allows him to pass freely between k^r and $w^r = MC^{-1}(k^r)$, and thus interchanging the order of the MixColumns and AddRoundKey operations does not change the situation.

There are also attacks that seem to be affected, but can be "rescued" by interchanging the order of the MixColumns and AddRoundKey operations in the second to last round. For example, in the improved attack on 7-round AES-128 presented in [9], the adversary can replace the second-to-last subkey k^5 with the equivalent subkey w^5 (which is possible since the adversary guesses two full columns of k^5). Then the adversary can use modified key relations like Equation (1) instead of the original relations.

However, as shown in the previous sections, there are classes of attacks for which the modified relations cannot be applied, and it seems that the removal of the last round MixColumns indeed reduces their complexity.

5 Summary and Conclusions

Our results show that the omission of MixColumns in the last round of AES reduces the security of reduced-round variants of AES with respect to various attacks. The security of the full AES may also be affected, if an attack on the full AES would use relations between the last round subkey and other subkeys.

We believe that our results raise the question whether the common practice of omitting the last round MixColumns in attacks on reduced-round AES is legitimate.

References

1. Behnam Bahrak, Mohammad Reza Aref, *A Novel Impossible Differential Cryptanalysis of AES*, proceedings of the Western European Workshop on Research in Cryptology 2007, Bochum, Germany, 2007.
2. Alex Biryukov and Dmitry Khovratovich, *Related-Key Cryptanalysis of the Full AES-192 and AES-256*, Advances in Cryptography, proceedings of ASIACRYPT 2009, Lecture Notes in Computer Science 5912, pp. 1–18, Springer, 2009.
3. Joan Daemen, Vincent Rijmen, *AES Proposal: Rijndael*, NIST AES proposal, 1998.
4. Joan Daemen, Vincent Rijmen *The design of Rijndael: AES — the Advanced Encryption Standard*, Springer-Verlag, 2002.
5. Orr Dunkelman and Nathan Keller, *Low Data Complexity Attacks on AES*, Early Symmetric Crypto (ESC) seminar, Remich, Luxembourg, January 2010. Available online at `https://cryptolux.org/mediawiki.esc/images/9/9b/LDC-AES.pdf`.
6. Hüseyin Demirci, Ali Aydin Selçuk, *A Meet-in-the-Middle Attack on 8-Round AES*, proceedings of Fast Software Encryption 15, Lecture Notes in Computer Science 5086, pp. 116–126, Springer-Verlag, 2008.
7. Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Mike Stay, David Wagner, Doug Whiting, *Improved Cryptanalysis of Rijndael*, proceedings of Fast Software Encryption 2000, Lecture Notes in Computer Science 1978, pp. 213–230, Springer-Verlag, 2001.
8. Henri Gilbert, Marine Minier, *A collision attack on 7 rounds of Rijndael*, proceedings of the Third AES Candidate Conference (AES3), pp. 230–241, New York, USA, 2000.
9. Jiqiang Lu, Orr Dunkelman, Nathan Keller, Jongsung Kim, *New Impossible Differential Attacks on AES*, proceedings of INDOCRYPT 2008, Lecture Notes in Computer Science 5365, pp. 279–293, Sprigner-Verlag, 2008.
10. Stefan Lucks, *Attacking Seven Rounds of Rijndael under 192-bit and 256-bit Keys*, proceedings of the Third AES Candidate Conference (AES3), pp. 215–229, New York, USA, 2000.
11. Raphael Chung-Wei Phan, *Impossible Differential Cryptanalysis of 7-round Advanced Encryption Standard (AES)*, Information Processing Letters, Vol. 91, Number 1, pp. 33-38, Elsevier, 2004.

12. US National Institute of Standards and Technology, *Advanced Encryption Standard*, Federal Information Processing Standards Publications No. 197, 2001.
13. Wentao Zhang, Wenling Wu, Dengguo Feng, *New Results on Impossible Differential Cryptanalysis of Reduced AES*, proceedings of ICISC 2007, Lecture Notes in Computer Science 4817, pp. 239–250, Springer-Verlag, 2007.