IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2017, No. 2, pp. 59–83. DOI:10.13154/tosc.v2017.i2.59-83

Human-readable Proof of the Related-Key Security of AES-128

Khoongming Khoo[1], Eugene Lee[2], Thomas Peyrin[3,4,5] and Siang Meng Sim[3]

[1] Defence Science Organisation (DSO) National Laboratories, Singapore, Singapore
kkhoongm@dso.org.sg
[2] Raffles Institution, Singapore, Singapore
bleh3@hotmail.com
[3] School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore, Singapore
crypto.s.m.sim@gmail.com
[4] School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore
[5] Temasek Laboratories, Nanyang Technological University, Singapore, Singapore
thomas.peyrin@ntu.edu.sg

Abstract. The related-key model is now considered an important scenario for block cipher security and many schemes were broken in this model, even AES-192 and AES-256. Recently were introduced efficient computer-based search tools that can produce the best possible related-key truncated differential paths for AES. However, one has to trust the implementation of these tools and they do not provide any meaningful information on how to design a good key schedule, which remains a challenge for the community as of today.
We provide in this article the first human-readable proof on the minimal number of active Sboxes in the related-key model for AES-128, without any help from a computer. More precisely, we show that any related-key differential path for AES-128 will respectively contain at least 0, 1, 3 and 9 active Sboxes for 1, 2, 3 and 4 rounds. Our proof is tight, not trivial, and actually exhibits for the first time the interplay between the key state and the internal state of an AES-like block cipher with an AES-like key schedule. As application example, we leverage our proofs to propose a new key schedule, that is not only faster (a simple permutation on the byte positions) but also ensures a higher number of active Sboxes than AES-128's key schedule. We believe this is an important step towards a good understanding of efficient and secure key schedule designs.

Keywords: AES · related-key differential attack · security proof · key schedule

1 Introduction

Block ciphers and more generally symmetric key cryptographic primitives are very important components in nowadays security systems. While one usually gets confidence in a symmetric-key ciphering scheme only after a thorough cryptanalysis period of several years, ensuring the soundness of the algorithm during the design phase is also primordial. In particular, it is today desired that newly proposed ciphers have very strong security arguments regarding state-of-the-art attacks. The most common security arguments put forward by designers is the resistance of the cipher against differential and linear cryptanalysis, that proved to be very powerful tools to break many primitives.

The symmetric key cryptography community has now reached a quite good experience in designing good permutations for which it is possible to provide such arguments. The

Received: 2016-09-01, Revised: 2017-03-01, Accepted: 2017-05-02, Published: 2017-06-19

current NIST block cipher standard AES [DR02] benefits from very simple yet powerful proofs on the minimal number of active Sboxes in a differential path (active/inactive Sbox refers to an Sbox containing/not containing a difference): it is very simple to prove that any differential path for AES will contain at least 25 active Sboxes over 4 rounds. These proofs are actually generic enough to be applied to other matrix sizes than the AES. From the minimal number of active Sboxes and the best differential transition through the Sbox, one directly deduces an upper bound on the probability of a differential path.

However, these AES security arguments only hold in the classical single-key security model in opposition to the related-key security model where the attacker is allowed to insert differences not only in the plaintext/ciphertext, but also in the key input of the block cipher. The related-key attacks are much harder to protect against, and many ciphers eventually got broken in this scenario. Even AES-192 and AES-256, respectively the 192-bit and 256-bit key versions of AES, were shown to be weak in this model [BK09, BKN09] (and possible patches of the key schedule were proposed in [Nik10, CZK+11]).

Matsui [Mat94] was the first to present an algorithm that searches for the best possible differential characteristics, and showed the soundness of such a strategy by applying it to the previous block cipher standard DES. Based on his work, we have recently seen the emergence of much more efficient computer assisted tools that can even work for the related-key model [BN10, BN11, MWGP11, FJP13, SHW+14] (unfortunately, they were introduced too late to avoid the issues on AES-192 and AES-256). These tools are generally smart exhaustive search through all the possible differential paths. As such, their efficiency is quite dependent on the parameters of the cipher analysed. This can be considered a drawback and while analysing AES-128 is definitely doable, reaching AES-256 is much harder and using a 8×8 matrix as internal state would probably render these strategies impossible to apply in practice. A second drawback is that one has to trust the implementation that has been utilized. The tool will output the best possible differential path that it found and because it was programmed to search exhaustively among all the candidates, we conclude that it is the best possible differential path. However, the mere knowledge of the best differential path found by the tool is not sufficient to be fully convinced, and one would have to review the entire code. Finally, the last drawback, and probably the most important, is that these tools don't tell us anything about how to design a good block cipher and in particular its key schedule component. No really meaningful information is output that can help the designers to understand the interactions going on between the key schedule part and the internal state part.

The problem of designing a good (i.e. secure and efficient) key schedule is actually very important and can also serve to incorporate tweak inputs in a block cipher [JNP14d]. Yet, it remains a difficult challenge for the designers as of today and no general construction has so far emerged as being particularly good. The classical strategy is to build an efficient key schedule function quite different from the internal state round function, in a hope that it will be difficult for the attacker to arrange good interactions between the key state and the internal cipher state (as for example in AES or in the PRESENT block cipher [BKL+07]). However, the obvious drawback of this strategy is that while it makes things hard for the cryptanalyst, it also makes it harder for anyone to prove or to get a good confidence in the security of the construction. Some primitives take a more security-based approach, but they suffer from efficiency loss. For example, the internal cipher of the WHIRLPOOL hash function [BR11] has a strong key schedule function that is almost the same as the internal state round function. This effectively ensures a good security in the related-key scenario, but basically doubles the number of computations required. On the other extreme, LED block cipher [GPPR11] has no key schedule at all and can trivially adapt the simple AES single-key security proofs to the related-key scenario, but at the cost of a significant increase in the number of rounds.

Our contributions

In this article, we provide the first human-readable proof on the minimal number of active Sboxes for 1/2/3/4 rounds of AES−128 in the related-key model, without any external computational help. Our bounds are tight regarding truncated differences.

Apart from the extra confidence that our proof provides to the AES−128 key schedule (as the current proofs on AES−128 were produced with search tools), we believe our work is a new step towards helping future designers to create efficient and secure key schedule algorithms. It provides new insight on the important and complex interactions between the key schedule state and the internal state in a AES-like cipher. As example of this new insight, leveraging our proofs, we also propose a new fully linear key schedule for AES-like function that is much faster than AES−128's key schedule (it is basically composed of only a simple permutation on the byte positions) and that guaranties more active Sboxes. The holy grail would be to generate an efficient key schedule algorithm for which the security against related-key attacks can be proven in a simple and generic way (i.e. for any parameters of the cipher).

We note that this is the first proof of this kind, as to the best of our knowledge all ad-hoc block ciphers that can prove resistance to related-key differential attacks used computer-aided proofs [SIH+11, Nik10, JNP14c, JNP14a, JNP14b] or non-efficient constructions [BR11, GPPR11, CZK+11, MHM+02].

In short, our article will be devoted to proving the following without any external computational help: *any non-null related-key differential path for 1, 2, 3 and 4 consecutive rounds of* AES−128 *contains at least 0, 1, 3 and 9 active Sboxes respectively.*

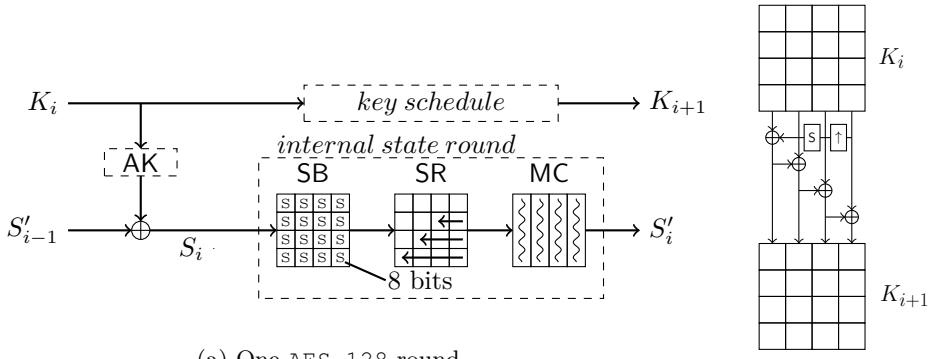
This article is organized as follows. Section 2 describes the structure of AES−128. Section 3 presents the notations and preliminary lemmas that will be used in the proofs. Section 4 and 5 provide the related-key security proofs for 1/2/3 and 4 consecutive rounds of AES−128 respectively. Section 6 discusses the design of a new key schedule that improves the related-key differential bounds. Section 7 concludes the article.

2 Description of AES−128

The AES−128 cipher takes as input a 128-bit plaintext and transforms it using a 128-bit key to produce a 128-bit ciphertext. The cipher is composed of 10 successive applications of a round function, and we denote by S_i, S'_i the internal input, output states of the round function, and by K_i the internal state of the key schedule at the i-th round. Each state can be viewed as a 4-by-4 array of bytes.

The AES−128 round is depicted in Figure 1a. At round i, AddRoundKey (AK) is performed to xor the previous output state S'_{i-1} with the round key K_i, to form input state S_i. This is transformed by the round function to produce output state S'_i. The round function first performs SubBytes (SB), which nonlinearly transforms each byte of the internal state by the AES Sbox. Then it does ShiftRows (SR), which shifts row r of the internal state by $(r-1)$-bytes to the left. Finally, MixColumns (MC) is applied, where each column is mixed through multiplication by an MDS matrix. The round key K_i is updated by a key schedule layer to form the next round key K_{i+1}. Please note that the MixColumns operation is omitted in the tenth round of AES−128 and a final round key is xored to produce the ciphertext.

The AES−128 key schedule function KS is given in Figure 1b. The key schedule internal state is a 4-by-4 byte array which is initialized by the 128-bit key. To compute the first column of K_{i+1}, the fourth column of K_i is upward rotated by 1-byte, and then transformed by the AES Sbox and xored with round constant RCON, before being xored to the first column of K_i. To compute subsequent columns $c = 2, 3, 4$ of K_{i+1}, column c of K_i is xored to column $c-1$ of K_{i+1}. Since round constant addition with RCON does not



(a) One AES-128 round.

(b) The AES-128 key schedule (omitting the round constant addition).

Figure 1: Overview of AES-128

affect differential cryptanalysis, we omit it in our subsequent analysis.

3 Preliminaries

3.1 Notation

In order to represent the various internal states and key state differences, we use the notations given in Figure 2. The initial state (respectively final state) of each round is denoted S_x (respectively S'_x), where x is the round number ($1 \leq x \leq 4$). The key state is denoted K_x.

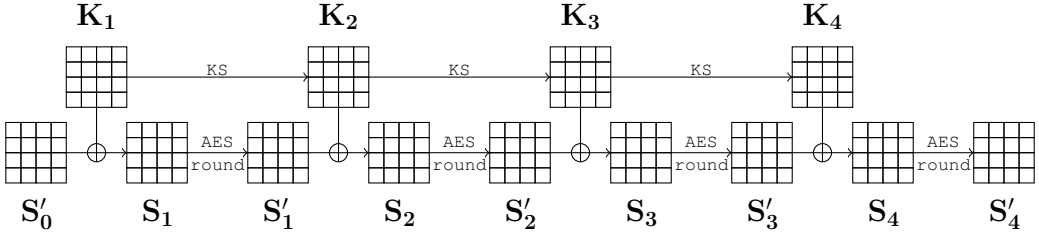


Figure 2: Notations for 4 rounds of AES-128.

Moreover, as depicted in Figure 3, we denote by $S^{|j}$ the byte column j (starting the counting from 1) of state S and similarly we use notation $S^{\backslash j}$ to represent the j-th top-left bottom-right byte diagonal (starting the counting from 1) of state S. More formally, for a state S, if one denotes $S^{i,j}$ the byte located at row i and column j (starting the counting from 1), $S^{|j}$ represents the four bytes $S^{i,j}$ with $i \in \{1,2,3,4\}$. Then, for the diagonal, we have $S^{\backslash 1} = \{S^{1,1}, S^{2,2}, S^{3,3}, S^{4,4}\}$, $S^{\backslash 2} = \{S^{1,2}, S^{2,3}, S^{3,4}, S^{4,1}\}$, $S^{\backslash 3} = \{S^{1,3}, S^{2,4}, S^{3,1}, S^{4,2}\}$ and $S^{\backslash 4} = \{S^{1,4}, S^{2,1}, S^{3,2}, S^{4,3}\}$. We will use interchangeably column j or j-th column to denote $S^{|j}$ (similarly diagonal j or j-th diagonal to denote $S^{\backslash j}$). Finally, in order to keep the notations light and since we will only deal with differences and not the values, we will also use S_x, K_x, etc. to denote the difference on S_x, K_x, etc. Then, $|S|$ (respectively $|S^{|i}|$ and $|S^{\backslash i}|$) will stand for the number of active bytes in a state S (respectively in a column $|S^{|i}|$ and diagonal $|S^{\backslash i}|$). A round x for which $|S_x| = |S'_x| = 0$ will be called a zero or inactive round. A column such that $|S^{|i}| = 4$ is called a fully active

column and in contrary if $|S^{|i}| = 0$ we call it a zero or inactive column. We naturally extend this column notation to diagonals.

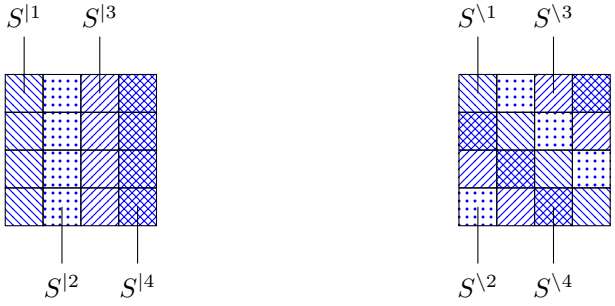


Figure 3: Column/diagonal notations $S^{|i}$ and $S^{\backslash i}$.

3.2 Best possible related-key differential paths

Before we begin our analysis on the differential characteristics in AES-128, we would like to exhibit an example of the best possible related-key truncated differential paths for AES-128, so that the readers can get a glimpse of the shape of such paths. This is done in Figure 4. We can observe that our example of the best path on 4 rounds actually contains an example of the best path on 3 rounds, which in turns contains an example of the best path on 2 rounds, etc.

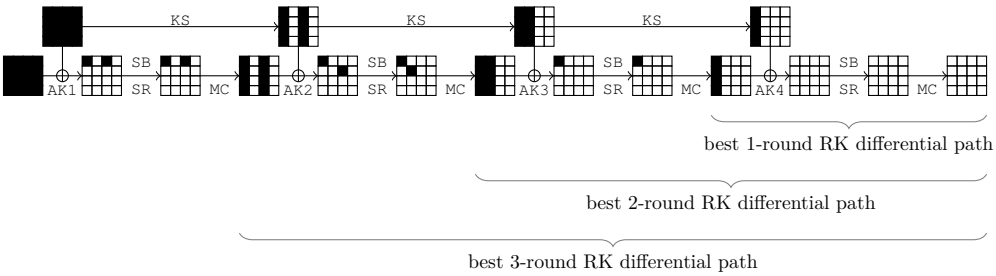


Figure 4: Example of best possible 1/2/3/4-round related-key truncated differential paths for AES-128. The black/white cells represent active/inactive bytes.

We emphasize that these characteristics are truncated, thus they don't consider the actual value inside the active bytes. This might lead to non-valid candidates as instantiating these differences might be impossible (due to the constraints imposed by the Sbox differential transitions and by the coefficients of the MixColumns layer). It turns out that this is actually the case here: the 4-round related-key path given in Figure 4 is impossible for AES-128. However, this path would potentially become possible if another diffusion matrix or another Sbox was utilised. Thus, these truncated differential paths tell a lot about the structural security provided by the AES-128.

We summarise in Table 1 the bounds on the number of active Sboxes proven by the computer-aided search tools [BN10, MWGP11, FJP13] in the related-key model and we compare them to our proven bounds. One can see that we achieve the same bounds. In other words, our bounds are tight. For completeness, we also provide the bounds proven by computer-aided search tools when the truncated differences must be instantiated to actual differences.

Table 1: Summary of the proven bounds on the number of active Sboxes in a related-key differential path for several rounds of AES-128. While previous works used computer-aided search, we obtained our bounds without any computation help.

Rounds	**1**	**2**	**3**	**4**
computer-aided bounds [BN10, FJP13] (non-truncated differences)	0	1	5	13
computed-aided bounds [BN10, MWGP11, FJP13] (truncated differences)	0	1	3	9
our bounds (truncated differences)	0	1	3	9

3.3 Key schedule patterns

Studying the related-key security of AES-128 regarding differential cryptanalysis will of course rely a lot on analysing how the truncated differences can spread through the key schedule. In Figure 5, we provide the truncated differential propagation of a single active byte, in both forward direction (key schedule function KS) and backward direction (inverse of the key schedule function $\mathtt{KS}^{-1}$). One can see that there are four different patterns obtained, depending on which column belongs the active byte in the starting key state K_x. All these patterns are invariant by rotation along the columns, so one directly obtains all the single active byte-induced patterns from this figure.

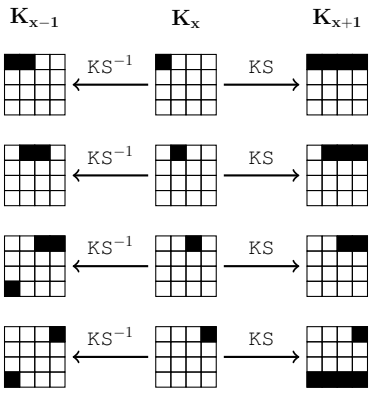


Figure 5: Backward/forward truncated differential transitions through the AES-128 key schedule when only a single byte is active in K_x. All these patterns are invariant by rotation along the columns. The black/white cells represent active/inactive bytes.

As the analysis depends heavily on the propagation (especially in the backward direction) of the active bytes through the key states, in Figure 6 we provide the detailed truncated differential propagation of a single active column, in both forward and backward direction. One can trace the propagation of each active cell in K_x through the same shade pattern. The white cells represent inactive bytes, while the gray cells can be active or inactive depending on the choice of active cells in K_x.

We emphasize that all these truncated differential transitions will happen with probability 1 (except for the bytes depicted in gray in Figure 6). This figure will be extensively referred to in our proof, as it will allow us to extract some properties on the truncated differential patterns when studying the various cases.

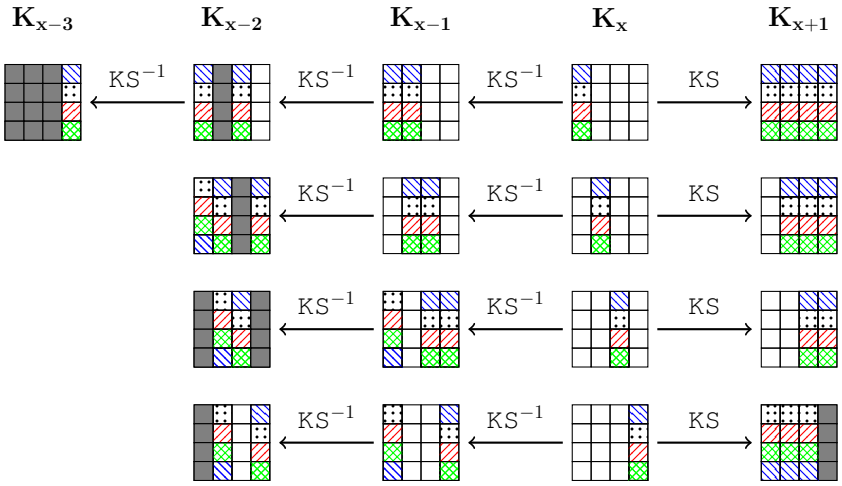


Figure 6: Backward and forward truncated differential transitions through the AES-128 key schedule when only one column is active in K_x. The propagation of each active cells in K_x can be traced using the same shade pattern. The white cells represent inactive bytes, while the gray cells can be either active or inactive.

3.4 Preliminary lemmas

3.4.1 Consecutive inactive rounds.

We first note that since we force a difference to be present in the key states, it is obviously impossible to have two consecutive inactive rounds.

Lemma 1. *It is impossible to have two consecutive inactive rounds when a difference is inserted in the key.*

Proof. Assume that $|S_x| = |S_{x-1}| = 0$. It implies that $|K_x| = 0$ which is impossible since every subkey must contain a difference (the AES-128 key schedule is a permutation). □

3.4.2 Adjacent key state bound.

We simply state here a simple bound due to the bitwise XOR operation during the key addition phase of an AES-128 round.

Lemma 2. *For any round x, we have $|S_x| \leq |S'_{x-1}| + |K_x|$, but also $|K_x| \leq |S_x| + |S'_{x-1}|$ and $|S'_{x-1}| \leq |K_x| + |S_x|$.*

Proof. Since XOR is a bitwise operation, and since $S_x = K_x \oplus S'_{x-1}$, we have that any active byte in S_x must necessarily come from an active byte in K_x or in S'_{x-1} (or both) at the same byte position. Thus, we directly deduce that $|S_x| \leq |S'_{x-1}| + |K_x|$ and the reasoning is identical for the two other inequalities. □

We remark that this lemma also holds if we take any column or diagonal separately. For example $|K_x^{|i}| \leq |S_x^{|i}| + |S'^{|i}_{x-1}|$ or $|K_x^{\backslash i}| \leq |S_x^{\backslash i}| + |S'^{\backslash i}_{x-1}|$ for any i.

3.4.3 MixColumns diffusion bound.

We state here the simple branching number of the matrix underlying the MixColumns diffusion layer of the AES-128 round.

Lemma 3. *For any round x, if a diagonal $S_x^{\backslash i}$ is active, then we have $|S_x^{\backslash i}| + |S_x'^{|i}| \geq 5$.*

Proof. Immediate since the branching number of the matrix underlying the MixColumns layer is equal to 5. □

3.4.4 Column sum bound.

We state here the relation between a subkey and two consecutive internal states columns or diagonals, due to the branching number of the AES-128 diffusion matrix. This lemma will be useful as it will give us a helpful bound on the number of active Sboxes given a certain subkeys truncated differential characteristic.

Lemma 4. *For any round x, we have $|S_{x-1}^{\backslash i}| + |S_x^{|i}| \geq 5 - |K_x^{|i}|$ if $|S_{x-1}'^{|i}| \neq 0$. Otherwise, we have $|S_{x-1}^{\backslash i}| = 0$ and $|S_x^{|i}| = |K_x^{|i}|$.*

Proof. First, if $|S_{x-1}'^{|i}| = 0$, we trivially deduce that $|S_{x-1}^{\backslash i}| = 0$ by inverting the AES-128 round function. Moreover, since there is no active byte in $S_{x-1}'^{|i}$, the active/inactive bytes pattern of $S_x^{|i}$ and $K_x^{|i}$ are exactly the same and thus $|S_x^{|i}| = |K_x^{|i}|$.

If $|S_{x-1}'^{|i}| \neq 0$, then the AES-128 diffusion matrix guaranties that $|S_{x-1}^{\backslash i}| \geq 5 - |S_{x-1}'^{|i}|$. Thus, $|S_{x-1}^{\backslash i}| + |S_x^{|i}| \geq 5 + |S_x^{|i}| - |S_{x-1}'^{|i}|$ and from the column version of Lemma 2 we deduce that $|S_{x-1}^{\backslash i}| + |S_x^{|i}| \geq 5 - |K_x^{|i}|$. □

3.4.5 Diffusion in the key schedule.

We give here a few useful lemmas regarding the diffusion of the AES-128 key schedule in the forward and backward direction.

Lemma 5. *For any round x, if K_x has only a single active column, then $|K_{x+1}| \geq 2 \cdot |K_x|$, $|K_{x-1}| \geq 2 \cdot |K_x|$ and $|K_{x-2}| \geq 2 \cdot |K_x|$. Moreover, at least two columns of K_{x+1}, K_{x-1} and K_{x-2} will contain exactly $|K_x|$ active bytes.*

Proof. This can directly be observed from Figure 6 by considering different combination of the active column. □

Lemma 6. *For any active row of K_x, if the rightmost active byte is in column c, where $1 \leq c \leq 4$, then for the same row in K_{x+c-4}, the byte in the 4th column will be active.*

Proof. One can observe from Figure 1b that the rightmost active byte will propagate (in the backward direction) one column to the right (unless it is already on the 4th column) without fail. Thus for any row with active byte, the rightmost active byte will propagate to the 4th column within 3 inverse key schedule function. □

Corollary 1. *For any four consecutive rounds x to $x+3$, if K_{x+3} has k active rows, then $\sum_{i=x}^{x+3} |K_i^{|4}| \geq k$.*

Proof. A direct consequence of Lemma 6. □

3.4.6 Internal inactive round bound.

The following lemma captures the intuition that if there is an inactive round at some point in the truncated differential path, then many Sboxes will be active in the previous and next round.

Lemma 7. *For any inactive round $x+1$ (i.e. $|S_{x+1}| = 0$), we have that $|S_x| + |S_{x+2}| \geq 6$.*

Proof. Suppose first that K_{x+1} contains only a single active column. From Lemma 5, one has that $|K_{x+2}| \geq 2 * |K_{x+1}|$. Then, since $|S_{x+1}| = |S'_{x+1}| = 0$, we have

$$\begin{aligned}
|S_x| + |S_{x+2}| &= |S_x| + |S_{x+1}| + |S'_{x+1}| + |S_{x+2}| \\
&\geq |S_x| + |S_{x+1}| + |K_{x+2}| && \text{(Lemma 2)} \\
&\geq 5 - |K_{x+1}| + |K_{x+2}| && \text{(Lemma 4)} \\
&\geq 5 + |K_{x+1}| && (|K_{x+2}| \geq 2 * |K_{x+1}|) \\
&\geq 6 && (|K_x| \neq 0 \text{ for all } x)
\end{aligned}$$

Suppose now that K_{x+1} contains more than a single active column. Thus, there must be a column u of K_{x+1} such that $|K_{x+1}^{|u}| > 0$ and a column $v > 1$ of K_{x+1} such that $|K_{x+1}^{|v}| > 0$. Since the round $x+1$ is inactive, we get that the active/inactive bytes pattern between $|S'_x|$ and $|K_{x+1}|$ are exactly the same (similarly, S_{x+2} and K_{x+2} also have the same pattern). Thus, there must be a column u of S'_x such that $|S_x'^{|u}| > 0$ and a column $v > 1$ of S'_x such that $|S_x'^{|v}| > 0$.

$$\begin{aligned}
|S_x| + |S_{x+2}| &= |S_x| + |K_{x+2}| && (S_{x+2} \text{ and } K_{x+2} \text{ have the same pattern}) \\
&\geq |S_x^{\setminus u}| + |S_x^{\setminus v}| + |K_{x+2}| && (|S_x| = \textstyle\sum_i |S_x^{\setminus i}|) \\
&\geq 1 + |S_x^{\setminus v}| + |K_{x+2}| && (|S_x^{\setminus u}| \neq 0) \\
&\geq 6 - |S_x'^{|v}| + |K_{x+2}| && \text{(Lemma 3)} \\
&= 6 - |K_{x+1}^{|v}| + |K_{x+2}| && (S'_x \text{ and } K_{x+1} \text{ have the same pattern}) \\
&\geq 6 - |K_{x+1}^{|v}| + |K_{x+2}^{|v-1}| + |K_{x+2}^{|v}| && (|K_{x+2}| = \textstyle\sum_i |K_{x+2}^{|i}|) \\
&\geq 6
\end{aligned}$$

The last inequality comes from the observation that $|K_{x+2}^{|v-1}| + |K_{x+2}^{|v}| \geq |K_{x+1}^{|v}|$. Indeed, since $v > 1$ the AES-128 key schedule tells us that $K_{x+2}^{|v-1} \oplus K_{x+2}^{|v} = K_{x+1}^{|v}$ and the inequality is obtained by a reasoning on the XOR operation just like for Lemma 2. □

From Lemma 7, one can draw some conclusions about the round key K_{x+1} when the bound is tight.

Corollary 2. *For any inactive round $x+1$ (i.e. $|S_{x+1}| = 0$), if $|S_x| + |S_{x+2}| = 6$, then the round key K_{x+1} either has*

- *exactly 1 active byte in the 3rd column,*
- *or exactly 2 fully active columns. In addition, $\sum_{i=x-1}^{x+2} |K_i^{|4}| \geq 4$.*

Proof. If $|S_x| + |S_{x+2}| = 6$, then all the inequalities in the proof of Lemma 7 must be tight. Hence, we reconsider the proof but with equalities and observe their implication.

Suppose first that K_{x+1} contains only a single active column. From the 5th equality, we see that $|K_{x+1}| = 1$, Figure 5 and the 4th equality imply that the active byte is in the 3rd column of K_{x+1}, else $|K_{x+2}| > 2 * |K_{x+1}|$.

Suppose now that K_{x+1} contains more than a single active column. The 2nd equality implies that K_{x+1} contains at most 2 active columns, column u and v. The 3rd equality implies $|S_x^{\setminus u}| = 1$, by the branching number we know that $K_{x+1}^{|u}$ is fully active. The 6th equality implies that K_{x+2} contains at most 2 active columns, column $v-1$ and v and from the last equality we see that $|K_{x+2}^{|v-1}| + |K_{x+2}^{|v}| = |K_{x+1}^{|v}|$. Now suppose $v < 4$, from Figure 1b, we see that $K_{x+1}^{|u}$ will propagate to $K_{x+2}^{|u}$ and is fully active. Hence from the previous equation, we must have $u = v - 1$ and $|K_{x+1}^{|v}| = 4$. On the other hand, if $v = 4$,

then $|K^{|1}_{x+2}| = 0$ only if $u = 1$ and $|K^{|v}_{x+1}| = 4$. Therefore, K_{x+1} has 2 fully active columns and by Lemma 6, $\sum_{i=x-1}^{x+2} |K_i^{|4}| \geq 4$. □

3.4.7 Impossible internal states.

In general, identifying the minimum number of active bytes in the internal states in the related-key scenario is much harder than the single-key scenario. Here we present a lemma that helps us to filter several cases where the internal states have little active bytes as such internal states are impossible.

Lemma 8. *If* $|S_x| \leq 1$ *and* $|S_{x+1}| \leq 1$*, then* $|S_{x+2}| > 3$*.*

Proof. If $|S_{x+1}| = 0$, by Lemma 7, $|S_x| + |S_{x+2}| \geq 6$, hence $|S_{x+2}| \leq 3$ is impossible. If $|S_x| = 0$, by Lemma 1, $|S_{x+1}| \neq 0$, this implies that K_{x+1} is single active byte while S'_{x+1} has a fully active column. Since $S_{x+2} = S'_{x+1} \oplus K_{x+2}$, one can observe from Figure 5 that $|S_{x+2}| \leq 3$ is impossible.

Finally for the case $|S_x| = |S_{x+1}| = 1$, since S_x is a single active byte, then S'_x is a single fully active column. Moreover, as S_{x+1} is a single active byte, $K_{x+1} = S'_x \oplus S_{x+1}$ must necessarily be either:

- a single fully active column,
- or 3 active bytes all located in the same column,
- or a single fully active column with an extra active byte located in another column.

For the first 2 cases, K_{x+2} has at least 2 active columns of 4 (or resp. 3) active bytes. Since S'_{x+1} is a single fully active column, $|S_{x+2}| \leq 3$ is impossible. For the last case, if the 3rd column is fully active, K_{x+2} is expected to have 2 fully active columns in the next round key, while the extra active byte will propagate and may cancel some active bytes in one row[1], but always adds an active byte in the 2nd column. Since S_{x+1} cancels at most 1 column of active bytes, there are at least 4 remaining active bytes (3 in a column and 1 in another column). If the fully active column is in the other columns, there will be more fully active columns and the similar argument holds. □

3.4.8 Round key bound under special internal state.

We give a few useful lemmas to better understand the lower bound of the number of active Sboxes in the round keys under some specific internal state.

Lemma 9. *For any four consecutive rounds* x *to* $x+3$*, if* $|S_{x+2}| = 1$ *and* $|S_{x+3}| = 0$*, then* $\sum_{i=x}^{x+3} |K_i^{|4}| \geq 4$*. In addition,* K_{x+1} *has at least 2 fully active columns.*

Proof. Since S_{x+2} is a single active byte and S_{x+3} is an inactive state, then $K_{x+3} = S'_{x+2}$ is a single fully active column. By Corollary 1, we have $\sum_{i=x}^{x+3} |K_i^{|4}| \geq 4$. By Lemma 5, K_{x+1} has at least 2 fully active columns. □

Lemma 10. *For any four consecutive rounds* x *to* $x+3$*, if* $|S_{x+2}| = 2$ *and* $|S_{x+3}| = 0$*, then either* $|K_{x+3}| = 3$ *and* K_{x+1} *will have at least 2 active columns of 3 active bytes, or* $\sum_{i=x}^{x+3} |K_i^{|4}| \geq 4$*.*

Proof. Since S_{x+2} has 2 active bytes, and S_{x+3} is an inactive state, then $K_{x+3} = S'_{x+2}$ must necessarily be either:

[1] plus a single byte in another row if the extra active byte is in the 4th column, but adds two active bytes in the 1st and 2nd columns, thus it is still impossible to reduce the number of active bytes in S_{x+2} to less than 4.

- a single fully active column,
- or 3 active bytes all located in the same column,
- or 2 fully active columns.

For the 2nd case, by Lemma 5, K_{x+1} has at least 2 active columns of 3 active bytes. For the other 2 cases, by Corollary 1, we have $\sum_{i=x}^{x+3} |K_i^{|4}| \geq 4$. □

Lemma 11. *For any four consecutive rounds x to $x+3$, if $|S_{x+2}| = |S_{x+3}| = 1$, then either $|K_{x+3}| = 3$ and K_{x+1} will have at least 2 active columns of 3 active bytes, or $\sum_{i=x}^{x+3} |K_i^{|4}| \geq 4$.*

Proof. Since S_{x+2} is a single active byte, then S'_{x+2} is a single fully active column. Moreover, as S_{x+3} is a single active byte, $K_{x+3} = S'_{x+2} \oplus S_{x+3}$ must necessarily be either:

- a single fully active column,
- or 3 active bytes all located in the same column,
- or a single fully active column with an extra active byte located somewhere else.

For the 2nd case, by Lemma 5, K_{x+1} has at least 2 active columns of 3 active bytes. For the other 2 cases, by Corollary 1, we have $\sum_{i=x}^{x+3} |K_i^{|4}| \geq 4$. □

4 Related-key security proof for 1/2/3 rounds of AES-128

Theorem 1. *Any non-null related-key differential path for 1, 2 and 3 consecutive rounds of* AES-128 *contains at least 0, 1, and 3 active Sboxes respectively.*

4.1 For 1 and 2 rounds of AES-128

The best related key differential paths for 1 and 2 rounds of AES-128 have 0 and 1 active Sboxes respectively and this is trivial to prove. For a single round, one simply inserts the same difference in the key and in the plaintext to get a zero difference coming the Sbox layer. For two rounds, Lemma 1 tells us that it is impossible to have two consecutive inactive rounds, and thus there must be at least one active Sbox in any of the two rounds. Thus, $|S_1| + |S_2| + |K_1^{|4}| + |K_2^{|4}| \geq |S_1| + |S_2| \geq 1$. The best 2-round path is very easy to build: force the top-left Sbox in S_1 to be the only active one and choose K_2 to have only its first column to be fully active, so as to fully correct the difference coming from $|S'_1|$ and ensure no active Sbox in S_2 ($|S_1| = 1$ and $|S_2| = 0$). Inverting the key schedule from K_2, one can check that K_1 will only have its two first columns active. Therefore, $|K_1^{|4}| = |K_2^{|4}| = 0$.

4.2 For 3 rounds of AES-128

The computer-aided search tools have shown that the best related-key differential paths contain 3 active Sboxes. Proving that it is impossible to obtain less than 3 active Sboxes in three consecutive AES-128 rounds is not so complicated once the Lemmas from Section 3.4 are introduced. First, one can observe that in order to have less than 3 active Sboxes, the three rounds obviously can't be all active. Moreover, Lemma 1 tells us that it is impossible to have two consecutive inactive rounds. Thus, we are left with five cases, $\{|S_1| = 0, |S_2| = 1, |S_3| = 1\}$, $\{|S_1| = 1, |S_2| = 0, |S_3| = 1\}$, $\{|S_1| = 1, |S_2| = 1, |S_3| = 0\}$, $\{|S_1| = 0, |S_2| = 2, |S_3| = 0\}$ and $\{|S_1| = 0, |S_2| = 1, |S_3| = 0\}$. Conveniently with Lemma 8, we are left with one case to be analysed, that is $\{|S_1| = 0, |S_2| = 2, |S_3| = 0\}$.

For this special case $\{|S_1| = 0, |S_2| = 2, |S_3| = 0\}$, one can observe that $K_2 = S_2$ and $K_3 = S'_2$ since S_1 and S_3 are both inactive rounds. Since $|S_2| = 2$, K_3 must necessarily be either:

- a single fully active column,
- or 3 active bytes all located in the same column,
- or 2 fully active columns.

However, one can see from Figure 5 that the key state K_2 with 2 active bytes will not propagate to any of the 3 patterns. Hence, this case is also not possible.

All these bounds are tight, as can be seen in Figure 4 where an example of best differential paths is given. One can note that all the best possible related-key differential paths on 1/2/3 rounds of AES-128 do not contain any active Sbox in the key schedule. This will be different for the 4-round case, which will render the analysis much more complex.

5 Related-key security proof for 4 rounds of AES-128

In this section, we would like to bound the minimal number of active Sboxes N_{SB} for four consecutive rounds of the AES-128, in the related-key model. More precisely, we will prove that at least 9 will be active for 4 consecutive rounds of AES-128 in the related-key model, matching the bound found by the computer-aided searches.

Theorem 2. *Any non-null related-key differential path for 4 consecutive rounds of* AES-128 *contains at least 9 active Sboxes.*

5.1 Structure of the proof

In the case where no difference is inserted in the key input, the classical AES proof can be applied and one can easily show that $N_{SB} \geq 25$. Therefore, we would like to prove $N_{SB} \geq 9$ when a non-zero difference is inserted in the key input (since the AES-128 key schedule is a bijection, we will have $|K_x| \neq 0$ for all x).

We note that the initial state S'_0, the final state S_5, and the final key K_5 do not need to be considered in our proof, since no Sbox will be applied to any of these states. Thus,

$$N_{SB} = \sum_{x=1}^{4} |S_x| + |K_x^{|4}|.$$

The proof will be decomposed in two steps. First we will show that one always has $\sum_{x=1}^{4} |S_x| \geq 5$ (see Section 5.2). In addition, in the event that the bound is tight, it is necessary that $\sum_{x=1}^{4} |K_x^{|4}| \geq 4$. This implies that for four consecutive rounds, we have either $N_{SB} \geq 9$ or $\sum_{x=1}^{4} |S_x| \geq 6$. For the latter case, $N_{SB} \geq 9$ when $\sum_{x=1}^{4} |K_x^{|4}| \geq 3$. Secondly, we will prove that if $\sum_{x=1}^{4} |K_x^{|4}| < 3$, then necessarily $N_{SB} \geq 9$ (see Section 5.3). From there we directly conclude that $N_{SB} \geq 9$. We provide a graphical depiction of the overall structure of the proof in Figure 7.

5.2 Internal state

In this section, we will show that there are always at least 5 active Sboxes in the internal state of 4 consecutive rounds of AES-128, i.e. $\sum_{x=1}^{4} |S_x| \geq 5$ (and in the event where $\sum_{x=1}^{4} |S_x| = 5$, we will show that $\sum_{x=1}^{4} |K_x^{|4}| \geq 4$). Lemma 7 directly tells us that when

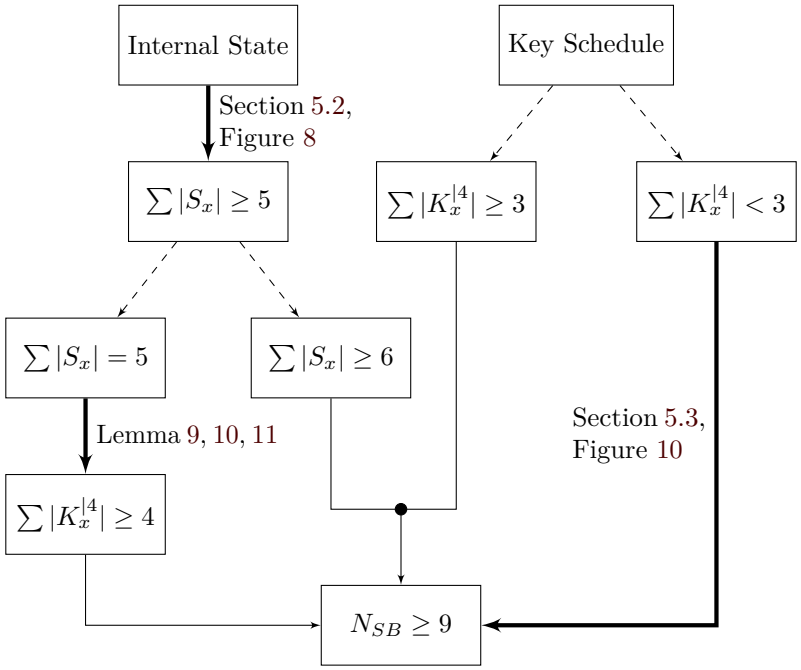


Figure 7: General structure of the proof of Theorem 2. A thick arrow represents a proven implication, a thin arrow represents a direct implication and hashed arrows represent subcases that are split.

the second or third rounds are inactive, then $\sum_{x=1}^{4} |S_x| \geq 6$. Thus, we only need to study the cases where both the second and third rounds are active (i.e. $|S_2| \neq 0$ and $|S_3| \neq 0$), and we will prove that all the configurations with $\sum_{x=1}^{4} |S_x| < 5$ are impossible. As there are several cases and subcases of internal state, we provide a graphical depiction of the overview of this section in Figure 8.

5.2.1 Case $|\mathbf{S_1}| = 0$.

Since S_1 is an inactive round, K_2 and S_2 will be equal.

- Let's first assume that $|S_2| = 1$, by Lemma 8, $|S_3| \leq 3$ is impossible. For $|S_3| \geq 5$, we have achieved $\sum_{x=1}^{4} |S_x| \geq 6$. Lastly for $|S_3| = 4$, since $S_2 = K_2$ with a single active byte, S_2' is a fully active column, the only possible case for $|S_3| = 4$ is when the single active byte is in the 3rd column of K_2 (see Figure 5) and S_3 has a column of 3 active bytes and a single active byte in a different column. This implies that S_3' has at least 2 fully active columns. On the other hand, K_4 has at most 6 active bytes as K_3 only has single active byte in the same row of the 3rd and 4th column. Therefore, S_4 is active and we have $\sum_{x=1}^{4} |S_x| \geq 6$.

- Let's now assume that $|S_2| = 2$, we show that it is always $\sum |S_x| \geq 6$. We denote a and b respectively the first and second active bytes of $S_2 = K_2$. If a and b are located in the same diagonal, then they will be involved in the same MixColumns function in the second round, in which case we obtain in $|S_2'|$ either 3 or 4 active bytes located in the same column (due to the MixColumns diffusion property). Otherwise, a and b will evolve in two different MixColumns functions and $|S_2'|$ will be composed of two fully active columns.

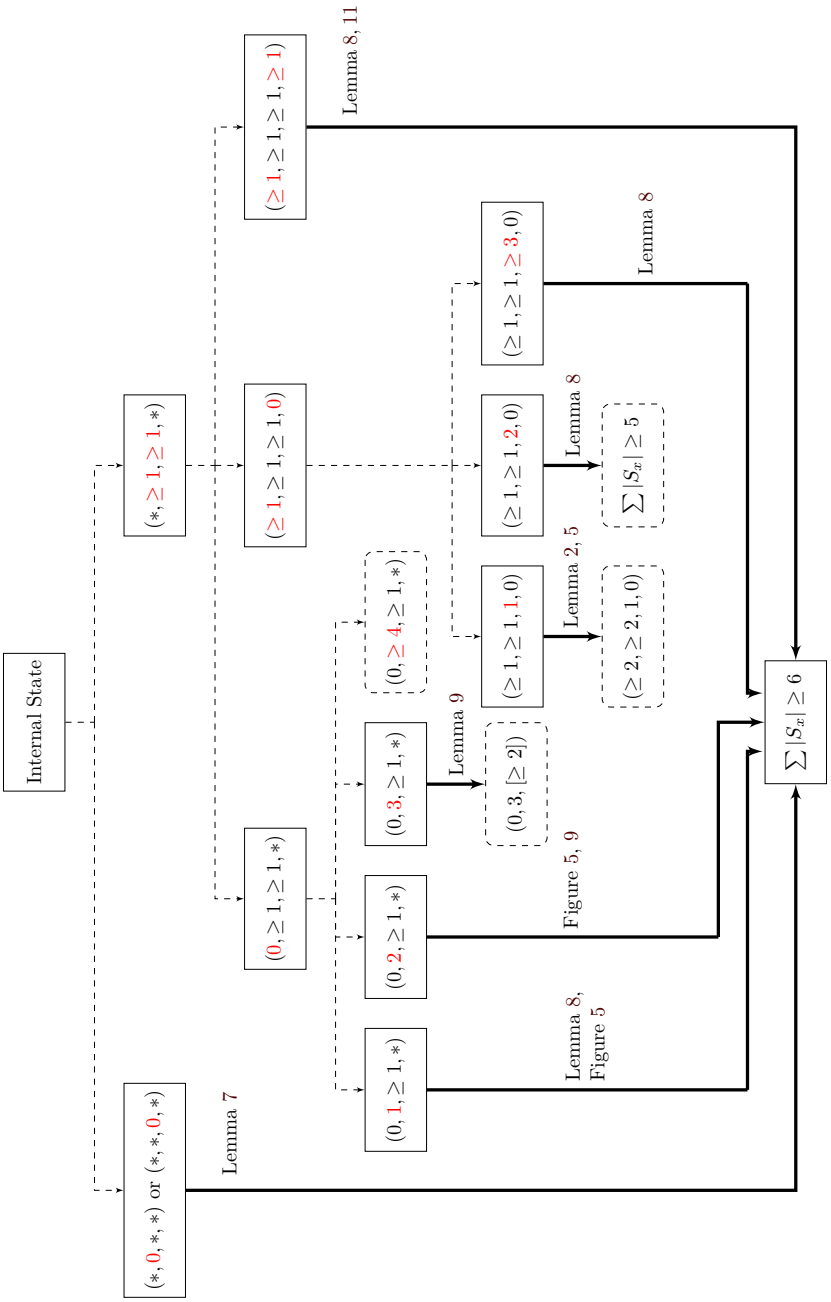


Figure 8: General structure of the proof in Section 5.2 for the internal state part. A thick arrow represents a proven implication, hashed arrows represent subcases that are split and a square parenthesis represents the sum of two states. A hashed box denotes the fact that it is subdivided into two subcases—$\sum |S_x| = 5$ and $\sum |S_x| \geq 6$.

– if $|S'_2| = 3$ (3 active bytes in the same column) or if $|S'_2| = 4$ (one fully active column). As in this scenario a and b must be in the same diagonal in $S_2 = K_2$, then it is one of the 6 possible cases (up to row rotations) from Figure 9,

one can see that $|S_3| \geq 4$ except for the last case where S_3 can have 3 active bytes, of which 2 active bytes are in the same row. However, in that case, after the MixColumns operation, S'_3 will have at least a fully active column but K_4 will have an inactive row, hence S_4 is active. Therefore, we always have $\sum_{x=1}^{4} |S_x| \geq 6$.

Figure 9: Key schedule patterns for two active bytes located in the same diagonal.

– if $|S'_2| = 8$ (two fully active columns). S_3 can be viewed as the XOR summation of S'_2, K_a and K_b, where the latter two are the forward propagation of a single active byte a and b, in another word, $K_a \oplus K_b = K_3$. Suppose a is in the fourth column, from Figure 5 one can see that K_a has a fully active row and 1 active byte in another row. Then $S'_2 \oplus K_a$ will have at least 7 active bytes (3 rows of 2 active bytes and 1 row of 1 active byte). Regardless the position of b, it can only erase up to 3 active bytes, hence we have $|S_3| \geq 4$. If b is in the fourth column, the same argument holds. Otherwise, if both a and b are not in the fourth column, there are at most 2 active rows in K_3, at best two bytes per column can be erased with K_3 and we have $|S_3| \geq 4$.

- Let's assume that $|S_2| = 3$. For at most 5 active Sboxes, we have three cases $\{|S_1| = 0, |S_2| = 3, |S_3| = 1, |S_4| = 0\}$, $\{|S_1| = 0, |S_2| = 3, |S_3| = 1, |S_4| = 1\}$ and $\{|S_1| = 0, |S_2| = 3, |S_3| = 2, |S_4| = 0\}$ since $|S_3| \neq 0$. Note that for all 3 cases, $K_2 = S_2$. By Lemma 9, the first case is impossible as K_2 should have at least 2 fully active columns which contradicts with $|S_2| = 3$. For the latter two cases with 5 active Sboxes, by Lemma 10 and 11 we have either $\sum_{x=1}^{4} |K_x^{|4}| \geq 4$, or K_2 has at least 2 active columns of 3 active bytes which contradicts with $|S_2| = 3$.

- Finally, let's assume that $|S_2| = 4$, the only possible tight $\sum_{x=1}^{4} |S_x| = 5$ case is $\{|S_1| = 0, |S_2| = 4, |S_3| = 1, |S_4| = 0\}$. By Lemma 9, we directly get $\sum_{x=1}^{4} |K_x^{|4}| \geq 4$ and we are done.

Note that cases where $|S_2| \geq 5$ can directly be discarded since $|S_3| \neq 0$, so we always have $\sum_{x=1}^{4} |S_x| \geq 6$.

5.2.2 Case $|S_1| \neq 0$ and $|S_4| = 0$.

Since S_4 is an inactive round, K_4 and S'_3 will be equal.

- Let's first assume that $|S_3| = 1$. After the application of the MixColumns function, $S'_3 = K_4$ will be a single fully active column. By Lemma 5, both K_2 and K_3 have at least 2 fully active columns. By Lemma 2, we have $|K_3| \leq |S_3| + |S'_2|$. If $|S_2| = 1$, then $|S'_2| = 4$ and the inequality does not hold. Thus, we must have $|S_2| \geq 2$. By

the same lemma, if $|S_1| = 1$, then $|K_2| \leq |S_2| + |S'_1|$ will imply that $|S_2| \geq 4$ and $\sum_{x=1}^{4} |S_x| \geq 6$. Finally for the case where $|S_1| \geq 2$ and $|S_2| \geq 2$, since we assumed $|S_3| = 1$ and $|S_4| = 0$, by Lemma 9, $\sum_{x=1}^{4} |K_x^{|4}| \geq 4$ and $N_{SB} \geq 9$.

- Let's now assume that $|S_3| = 2$. The only possible configuration for $\sum_{x=1}^{4} |S_x| < 5$ is $\{|S_1| = 1, |S_2| = 1, |S_3| = 2, |S_4| = 0\}$, which is impossible by Lemma 8. Hence, we have $\sum_{x=1}^{4} |S_x| \geq 5$. Since we have $|S_3| = 2$ and $|S_4| = 0$, by Lemma 10, we have either $\sum_{x=1}^{4} |K_x^{|4}| \geq 4$, or K_2 has at least 2 active columns of 3 active bytes. For the latter case, suppose that $|S_1| = 1$, then S'_1 is a fully active column and $S_2 = S'_1 \oplus K_2$ has at least one active column of 3 active bytes, hence $\sum_{x=1}^{4} |S_x| \geq 1 + 3 + 2 = 6$. Suppose $|S_1| = 2$, then S'_1 is either an active column (same conclusion as before) or 2 fully active columns, in which case S_2 at least 2 active bytes, hence $\sum_{x=1}^{4} |S_x| \geq 2 + 2 + 2 = 6$. For $|S_1| \geq 3$, we immediately have $\sum_{x=1}^{4} |S_x| \geq 3 + 1 + 2 = 6$.
- Finally, let's assume that $|S_3| = 3$, the only possible tight case is $\{|S_1| = 1, |S_2| = 1, |S_3| = 3, |S_4| = 0\}$. By Lemma 8, this is impossible.

Note that cases where $|S_3| \geq 4$ can directly be discarded since $|S_1| \neq 0$ and $|S_2| \neq 0$, so we always have $\sum_{x=1}^{4} |S_x| \geq 6$.

5.2.3 Case $|S_1| \neq 0$ and $|S_4| \neq 0$.

In this subpart, we study the event where all internal states are active ($|S_i| \geq 1$ for all $1 \leq i \leq 4$). There are five cases where $\sum_{x=1}^{4} |S_x| \leq 5$, namely $\{|S_1| = 1, |S_2| = 1, |S_3| = 1, |S_4| = 1\}$, $\{|S_1| = 1, |S_2| = 1, |S_3| = 1, |S_4| = 2\}$, $\{|S_1| = 1, |S_2| = 1, |S_3| = 2, |S_4| = 1\}$, $\{|S_1| = 1, |S_2| = 2, |S_3| = 1, |S_4| = 1\}$ and $\{|S_1| = 2, |S_2| = 1, |S_3| = 1, |S_4| = 1\}$. By Lemma 8, the only case that is left possible is $\{|S_1| = 1, |S_2| = 2, |S_3| = 1, |S_4| = 1\}$. By Lemma 11, either we are done or K_2 has at least 2 active columns of 3 active bytes. Since S'_1 has exactly a fully active column, it is not possible for $|S_2| = 2$. Thus for all four states to be active, $\sum_{x=1}^{4} |S_x| \leq 5$ is impossible and we have $\sum_{x=1}^{4} |S_x| \geq 6$.

5.3 Key schedule

In this section, we show that if $\sum_{x=1}^{4} |K_x^{|4}| < 3$, then necessarily $N_{SB} \geq 9$. In Figure 10, we provide a graphical depiction of the overview of this section.

Before going on, we make the observation that the difference propagation of four consecutive round keys is uniquely defined by their last columns. Precisely, if we let the differential values in the last column of round keys K_1, K_2, K_3, K_4 be A, B, C, D, then we can solve for every column of all four round keys via the `AES-128` key schedule equation (see Section 2).

This is shown in Figure 11 where N represents $A \oplus B \oplus C \oplus D$ and A', B' and C' represent the differential values of A, B and C after being rotated upward and sent through the `AES` S-box (the truncated difference column patterns A', B' and C' are just rotated versions of A, B and C respectively). A column with several variables in them represents an XOR-sum. For example, the first column of K_1 can be computed as $N \oplus A' \oplus B' \oplus C'$.

This gives us useful information by splitting cases on A, B, C and D, because they are the only key schedule columns that contribute to active S-boxes (the Sbox is applied only to the last column in `KS` - see Figure 1b). Since we are considering $\sum_{x=1}^{4} |K_x^{|4}| < 3$, we look at the cases where there are 0, 1 or 2 active bytes among the columns A, B, C and D.

For the rest of this section, whenever we assume $|S_3| = 0$, by Lemma 7 we have $|S_2| + |S_4| \geq 6$. By Corollary 2, having a tight bound implies that K_3 is either has

- exactly 1 active byte in the 3rd column,

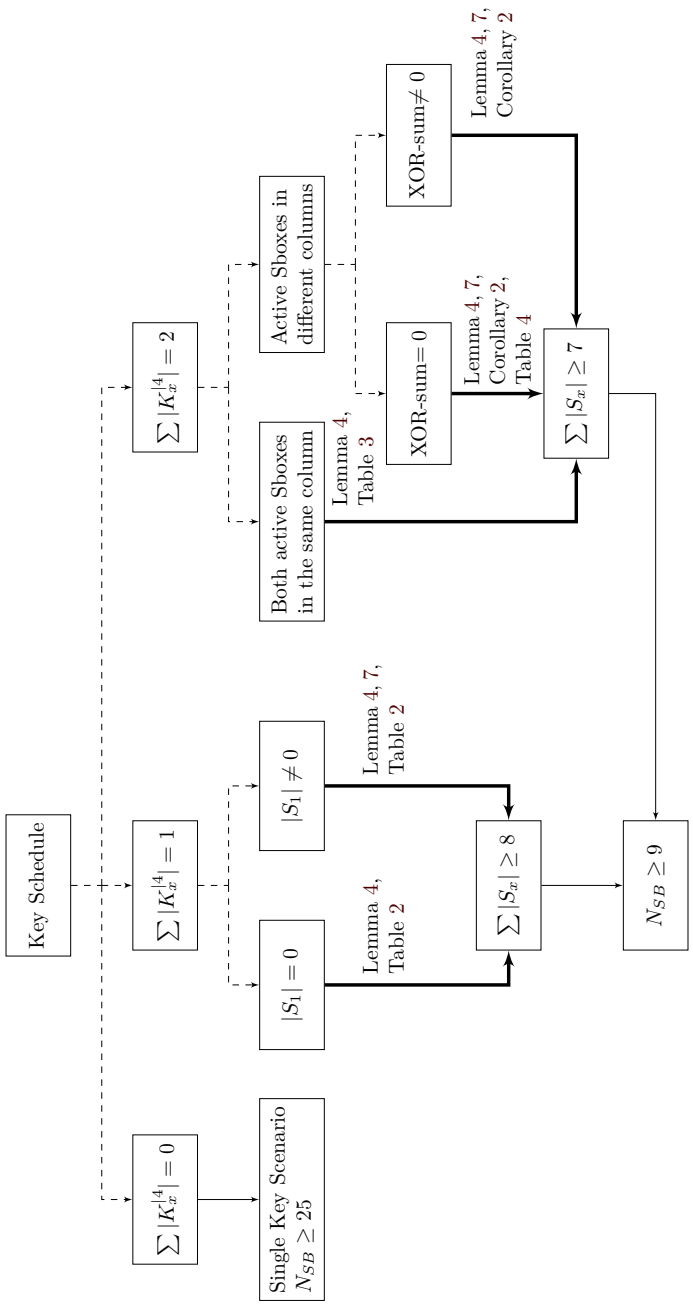


Figure 10: General structure of the proof in Section 5.3 that $\sum |K_x^{|4}| < 3$ implies $N_{SB} \geq 9$. A thick arrow represents a proven implication, a thin arrow represents a direct implication and hashed arrows represent subcases that are split.

- or exactly 2 fully active columns. In addition, $\sum_{x=1}^{4} |K_x^{|4}| \geq 4$.

Since we are considering $\sum_{x=1}^{4} |K_x^{|4}| < 3$, we only need to check if K_3 meets the first description, else we conclude that the bound cannot be tight and we have $|S_2| + |S_4| \geq 7$.

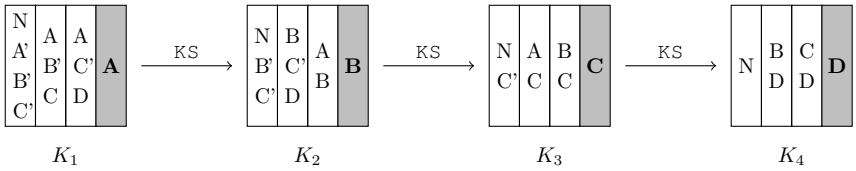


Figure 11: 4-round difference propagation of AES-128 key schedule.

5.3.1 Zero active Sbox in key schedule.

This case means that A, B, C and D all have zero difference, which implies that the entire key schedule has zero active bytes (see Figure 11). This contradicts our original assumption that a non-zero difference is inserted in the key input of the cipher.

5.3.2 One active Sbox in key schedule, occurring in any of the columns A, B, C and D.

By studying Figure 11 (see also Table 2), we see that an active byte in columns A, B, C or D contributes at most 1 active byte to every column of the key schedule, except the first column of K_1, K_2 and K_3 which may contain 2 active bytes (propagated from active last column). Also, we can see from Figure 11 that K_2 contains at least 2 active bytes on the same row.

Table 2: Number of active bytes in every column of K_2, K_3, K_4 when we have a single active byte in A or B or C or D.

Case	K_2	K_3	K_4
$\|A\| = 1$	$1, 0, 1, 0$	$1, 1, 0, 0$	$1, 0, 0, 0$
$\|B\| = 1$	$2, 1, 1, 1$	$1, 0, 1, 0$	$1, 1, 0, 0$
$\|C\| = 1$	$2, 1, 0, 0$	$2, 1, 1, 1$	$1, 0, 1, 0$
$\|D\| = 1$	$1, 1, 0, 0$	$1, 0, 0, 0$	$1, 1, 1, 1$

- Suppose $|S_1| = 0$. Then $S_2 = K_2$ and hence it contains at least 2 active bytes on the same row. Thus, S_2' contains at least two active columns u, v. By Lemma 4, we have:

$$\begin{aligned}\sum_{x=1}^{4} |S_x| &\geq |S_2| + |S_3| \geq (|S_2^{\setminus u}| + |S_3^{|u}|) + (|S_2^{\setminus v}| + |S_3^{|v}|) \\ &\geq (5 - |K_3^{|u}|) + (5 - |K_3^{|v}|) \geq 3 + 4 = 7,\end{aligned}$$

  because there is at most one column of weight 2 in K_3 while the rest are at most weight 1. This bound is tight only if $|K_3^{|u}| = 2$, $|K_3^{|v}| = 1$ and the other 2 columns to be inactive. However there is no such K_3, hence we have at least 8 active Sboxes in the main cipher together with 1 active Sbox in the key schedule, which gives $N_{SB} \geq 9$ and we are done.

- Suppose $|S_1| \neq 0$. If $|S_3| = 0$, then by Lemma 7, we have $|S_2| + |S_4| \geq 6$. The bound is not tight since there is no such K_3 with a single active byte in the 3rd column. Hence we have $|S_2| + |S_4| \geq 7$ which implies $\sum_{x=1}^{4} |S_x| \geq 8$ because $|S_1| \neq 0$. These 8 active Sboxes in the main cipher together with 1 active Sbox in the key schedule, which gives $N_{SB} \geq 9$ and we are done. Otherwise, if $|S_3| \neq 0$, we have at least one

active column u in S'_1, one active column v in S'_3 and note that K_2 has at least 2 active columns. By Lemma 4, we have:

$$\sum_{x=1}^{4} |S_x| \geq (5 - |K_2^{|u}|) + \min[(5 - |K_2^{|w}|), |K_2^{|w}|] + (5 - |K_4^{|v}|) \geq 3 + 1 + 4 = 8,$$

because there are at most 2 active bytes in each column of K_2 and at most 1 active byte in each column of K_4. Thus we are done as before because we have 8 active Sboxes in the main cipher.

Table 3: Number of active bytes in every column of K_2, K_3, K_4 when we have two active bytes in A or B or C or D, where n is either 3 or 4.

Case	K_2	K_3	K_4
$\|A\| = 2$	$2, 0, 2, 0$	$2, 2, 0, 0$	$2, 0, 0, 0$
$\|B\| = 2$	$n, 2, 2, 2$	$2, 0, 2, 0$	$2, 2, 0, 0$
$\|C\| = 2$	$n, 2, 0, 0$	$n, 2, 2, 2$	$2, 0, 2, 0$
$\|D\| = 2$	$2, 2, 0, 0$	$2, 0, 0, 0$	$2, 2, 2, 2$

5.3.3 Two active Sboxes in key schedule, occurring in one column of A, B, C and D.

By studying Figure 11 (see also Table 3), we can verify that if one of the columns of A, B, C or D contains 2 active bytes, then there will be at least two columns u, v in K_2 and one column w in K_4 with 2 active bytes. Hence by Lemma 4, we have:

$$\begin{aligned}
\sum_{x=1}^{4} |S_x| &\geq (|S_1^{\setminus u}| + |S_2^{|u}|) + (|S_1^{\setminus v}| + |S_2^{|v}|) + (|S_3^{\setminus w}| + |S_4^{|w}|) \\
&\geq \min[(5 - |K_2^{|u}|), |K_2^{|u}|] + \min[(5 - |K_2^{|v}|), |K_2^{|v}|] + \min[(5 - |K_4^{|w}|), |K_4^{|w}|] \\
&\geq 2 + 2 + 2 = 6,
\end{aligned}$$

Note that this inequality can be tight only if $|S_3| = 0$, otherwise by Lemma 4, the third term will be at least 3 and we get at least 7 active Sboxes in the main cipher. Again, since no such K_3 with a single active byte in the 3rd column, there are at least 7 active Sboxes in the main cipher together with the 2 active Sboxes in the key schedule, which gives $N_{SB} \geq 9$ and we are done.

Table 4: Number of active bytes in every column of K_2, K_3, K_4 when we have a two active bytes in A/B or A/C or A/D or B/C or B/D or C/D with XOR-sum = 0, where n is either 0 or 1.

Case (XOR-sum = 0)	K_2	K_3	K_4
$\|A\| = \|B\| = 1$	$1, 1, 0, 1$	$0, 1, 1, 0$	$0, 1, 0, 0$
$\|A\| = \|C\| = 1$	$1, 1, 1, 0$	$1, 0, 1, 1$	$0, 0, 1, 0$
$\|A\| = \|D\| = 1$	$0, 1, 1, 0$	$0, 1, 0, 0$	$0, 1, 1, 1$
$\|B\| = \|C\| = 1$	$n, 2, 1, 1$	$1, 1, 0, 1$	$0, 1, 1, 0$
$\|B\| = \|D\| = 1$	$1, 0, 1, 1$	$0, 0, 1, 0$	$0, 0, 1, 1$
$\|C\| = \|D\| = 1$	$1, 2, 0, 0$	$1, 1, 1, 1$	$0, 1, 0, 1$

5.3.4 Two active Sboxes in key schedule, distributed among two columns of A, B, C and D.

We split the proof into two parts, on whether the two active columns cancel each other.

- Suppose the two active columns have XOR-sum = 0. Then from Figure 11 (see also Table 4), we see that each column of K_4 has at most one active byte. If $|S_3| = 0$, then by Lemma 7, we have $|S_2| + |S_4| \geq 6$. By Corollary 2, the tight case implies that column B and D have to be active for K_3 to meet the first description. Since $S'_2 = K_3$ has 1 active byte, S_2 has a fully active diagonal. However, since K_2 for this case has its 2nd column inactive, S'_1 is active. Therefore, there are at least 7 active Sboxes in the main cipher, which together with the 2 active Sboxes in the key schedule gives $N_{SB} \geq 9$ and we are done. Otherwise S'_3 has an active column u and by Lemma 4, we have:
$$|S_3| + |S_4| \geq 5 - |K_4^{|u}| \geq 4,$$
because each column of K_4 has at most 1 active byte. We can also verify from Figure 11 that K_2 contains at least two active columns, which implies $|S_1| + |S_2| \geq 2$ by Lemma 4. Having this bound to be tight implies two things, S_1 is inactive and K_2 contains exactly two active columns of 1 active byte, which can only be the case where columns A and D are active. Since $S_2 = K_2$ with 2 active bytes in the same row (as $A = D$), $|S'_2| = 8$ while $|K_3| = 1$ implies that $|S_3| \geq 7$. Lastly, when the bound is not tight, $\sum_{x=1}^{4} |S_x| \geq 3 + 4 = 7$ active Sboxes in the main cipher, which together with the 2 active Sboxes in the key schedule gives $N_{SB} \geq 9$ and we are done.

- Suppose the two active columns have XOR-sum $\neq 0$. Then by studying Figure 11, we can verify that every column of K_4 has at most 2 active bytes, and that K_2 and K_4 have at least 5 active columns between them. Suppose $|S_3| = 0$, then by Lemma 7, we have $|S_2| + |S_4| \geq 6$. Once again by Corollary 2, the tight case is impossible as no K_3 fits the first description[2]. With at least 7 active Sboxes in the main cipher, which together with the 2 active Sboxes in the key schedule gives $N_{SB} \geq 9$ and we are done. Otherwise S'_3 has an active column u and by Lemma 4, we have:
$$|S_3| + |S_4| \geq 5 - |K_4^{|u}| \geq 3,$$
because each column of K_4 has at most 2 active bytes. Besides this column, there are at least 4 more active columns in K_2 and K_4, which by Lemma 4 ensures at least 4 more active Sboxes in the main cipher. Thus in total we get at least 7 active Sboxes in the main cipher, which together with the 2 active Sboxes in the key schedule gives $N_{SB} \geq 9$ and we are done.

6 On designing better key schedules for AES-128

As an example of the insight that our proofs provide on the interplay between the internal state function and the key schedule, we propose a new fully linear key schedule that can be used to replace the one in AES-128. Our new key schedule proposal is simple: it is basically a permutation on the key state byte positions. More precisely, the key state update function will simply:

- rotate respectively by $(1, 0, 0, 2)$ positions to the right the bytes located in the $(1, 2, 3, 4)$-th row of the key state matrix

[2] The case where column B and D are active may seem to fit the first description of Corollary 2, but since XOR-sum $\neq 0$, the first column is active. Thus, it does not meet the first description.

- rotate the entire key state matrix by one position down

In other words, our key schedule proposal simply applies the following permutation on the bytes positions of the 4×4 key state matrix:

$$\begin{pmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{pmatrix} \longrightarrow \begin{pmatrix} 11 & 15 & 3 & 7 \\ 12 & 0 & 4 & 8 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \end{pmatrix}$$

Note that this key schedule is extremely efficient when compared to the original AES−128 key schedule. Not only it doesn't use any Sbox layer, but it also doesn't use any XOR for diffusion. In particular, a byte permutation is very well suited for hardware implementations at it merely consists in wiring, and one can also hard-code the entire key for further efficiency gain on the area used to store the key bits.

Regarding security, even though we don't use any Sbox nor XOR in our key schedule, we can actually prove more active Sboxes in the related-key model than for AES−128 as shown in Table 5 (note that we only obtained lower bounds for new key schedule, so the bounds might be even better than depicted in the table). These lower bounds on the number of active Sboxes, especially the bounds for the high number of rounds, were computed with computer-aided tools. However, the design of the key schedule directly comes from an analysis of our proofs.

Table 5: Summary of the proven lower bounds on the number of active Sboxes in a related-key differential path for several rounds of AES−128 with the key schedule replaced with our candidate.

Rounds	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
AES−128 key schedule (truncated differences)	0	1	3	9	11	13	15	21
our new key schedule (truncated differences)	0	1	5	10	14	18	21	25

The key schedule proposed in [Nik10] requires 32-bit XORs and 4 Sboxes, achieving related-key differential bounds of 1, 5, 10 and >11 active Sboxes for 2, 3, 4 and 5 rounds respectively. Note that these include the active Sboxes in the key schedule. On the other hand, our proposal does not require any XORs or Sboxes but yet already achieving 1, 5, 10 and 14 active Sboxes respectively.

Our strategy to build that key schedule was to look at our proof for 3 rounds (in order to start with a simple proof) and analyse what modifications in the design could improve the minimal number of active Sboxes (only 3 in the case of AES−128's key schedule). First, since no active Sbox will be involved in the key schedule state for the best 3-round related-key differential paths on AES−128, one could actually directly remove the Sbox layer in the key schedule without impacting the bound. Secondly, by removing the XORs in the key update function, our goal was to reach a zero branching which will make the analysis simpler (the tracking effort in the proof is reduced): the number of active bytes in the key state always remains constant throughout the rounds.

We ended up with the key update function being simply a permutation on the key byte positions (and not on the key bit positions themselves, as it would make the security analysis very complex and software implementations very inefficient). One can see from our proofs that a lot of complex cases arise during the analysis because some difference introduced by a subkey can be corrected by the next (forward) or previous (backward) subkey. The choice of the permutation was thus done with the following criteria in mind:

we tried to minimise the overlap[3] between KS and MC ∘ SR ∘ SB and between $\texttt{KS}^{-1}$ and $\texttt{SB}^{-1} \circ \texttt{SR}^{-1} \circ \texttt{MC}^{-1}$. This criterion actually makes a lot of sense, but was never taken into account in previous key schedule designs.

Note that several permutations and different choices than $(1, 0, 0, 2)$ right rotate verify this, but if the sum of the right rotate values is coprime with 4, then a key byte would go through all 16 possible positions as the key schedule iterates (which is a desirable design property). Finally, $(1, 0, 0, 2)$ was the simplest choice that follows these two criteria.

Proving in the related-key model at least 3 active Sboxes for 3 rounds is trivial for this key schedule: Lemma 1 tells us that it is impossible to have two consecutive inactive rounds. Thus, as for AES-128 key schedule, for $\sum_{x=1}^{4} |S_x| < 3$ we are left with five cases, $\{|S_1| = 0, |S_2| = 1, |S_3| = 1\}$, $\{|S_1| = 1, |S_2| = 0, |S_3| = 1\}$, $\{|S_1| = 1, |S_2| = 1, |S_3| = 0\}$, $\{|S_1| = 0, |S_2| = 2, |S_3| = 0\}$ and $\{|S_1| = 0, |S_2| = 1, |S_3| = 0\}$. It is very easy to see that the first, second, fourth and fifth cases are impossible ($|S_x| = 0$ and $|S_{x+1}| = u$ means that all subkeys have u active bytes, which contradicts the internal state constraints for all four cases). Finally, for the third case, $|S_2| = 1$ and $|S_3| = 0$ means that all subkeys have 4 active bytes because S'_2 is a fully active column. Yet, thanks to the design criterion of the key schedule, the active byte in S_2 can't be erased backward with the subkey active bytes, and thus $|S_1| \geq 4$ which is a contradiction.

One could naturally tweak this design (without increasing the tracking effort) by adding an Sbox layer every round to the entire first row of the key state. Because of the down rotation in the key update function, we would be ensured that at least one Sbox will necessarily be active in 4 consecutive key states. In addition, the number of active bytes in the key state remains the same, hence we can easily see that in 4 consecutive rounds, the number of active bytes in the key state will result in the same number of active Sboxes in the key schedule. This would directly add to the lower bounds given in Table 5.

We emphasize that our goal here is to design a simple yet efficient key schedule that maximizes the number of active Sboxes in the related-key model. Of course, other security considerations than the number of active Sboxes should be taken into account when designing a cipher, as the key schedule impacts many types of attacks. In particular, one might be tempted to believe that having no diffusion or Sboxes in the key schedule will result in the cipher being susceptible to key recovery attack like the Meet-in-the-Middle (MitM) attack. Using the open source code provided by Derbez [DFJ13, DF16], we analysed and found that MitM attack can reach up to 8 rounds of AES under this new key schedule while the same attack can reach up to 7 rounds of AES-128. One can thus observe that using a linear key schedule with no diffusion does not necessarily severely weaken its resistance against key recovery attack like MitM attack. In addition, this new key schedule is more efficient in area, throughput (no Sbox to compute in the key schedule), and provides better related-key differential bounds. Besides, we argue that it is a better strategy as a designer to reduce the attack surface that leads to a key recovery attack (for example by forcing a higher number of active Sboxes), than just trying to make a key recovery harder when such flaw is present.

7 Discussions and future works

Our work leads to many interesting future directions and open problems. First, it would be interesting to look at what happens for 5 rounds, 6 rounds, etc. Indeed, the best related-key truncated differential path from Figure 4 (like any 4-round path with this amount of active Sboxes) is not iterative and one can hope for a better bound than $9 + 0 = 9$ (9 active Sboxes for 4 rounds and 0 for 1 round).

[3] By "overlap" we meant that if the internal cipher and the key schedule are well synchronized, it is likely that good differential paths exist (see for example the attacks on the full Whirlpool hash function).

A second topic to explore is the scalability of our proof to higher dimensions: can we prove anything when using bigger matrix sizes? This is actually quite an interesting direction, because the computational complexity of the automated tools grows very fast with the size of the internal state or key state matrix. For example, what would be the best possible related-key truncated differential path of an AES-like primitive with a 8×8 matrix size? Lightweight encryption schemes are likely to use smaller Sboxes and thus likely to have bigger matrices to handle. Moreover, can our proofs easily be adapted to bigger key sizes, such as AES-192 and AES-256?

A more difficult goal would be to obtain a proof on the best related-key differential path for AES-128, with actual differences instead of truncated differences. Indeed, while our bounds apply to the general structure of AES-128, we remark for example that the best related-key truncated differential path from Figure 4 is actually impossible to instantiate, due to some constraints coming from the coefficients of the MDS matrix in the MixColumns function. Looking at Table 1, we can see that there is a lot of room for improvements. Yet, we believe this would not be trivial to achieve, at least if the aim is to obtain tight bounds, as incorporating the MDS coefficients inside the proof looks difficult. However, improving a little the bounds we provide here looks feasible and such improvement would be very meaningful since the minimal number of active Sboxes we have proven does not directly place AES-128 out of target against related-key attacks: since the best differential transition probability of the AES Sbox is 2^{-6} and since we can prove $9 + 9 + 1 = 19$ active Sboxes on the full 10-round AES-128 (9 active Sboxes for 4 rounds and 1 for 2 rounds), one can only conclude that any related-key differential path will have a probability smaller than 2^{-114} ($> 2^{-128}$).

Finally, the most promising direction is constructive and extends the example we proposed in Section 6: can we design an efficient key schedule for AES-128, such that very good bounds on the number of active Sboxes in the related-key model can be guaranteed, with a simple and clean proof (even though it represents an improvement over the state-of-the-art, our key schedule from Section 6 suffers from the drawback that proving its security for many rounds still requires the use of computed-aided proofs). Can we adapt the AES-128 key schedule and hopefully remove the bottleneck parts of our proofs regarding the number of active Sboxes? In particular, it seems that the best related-key truncated differential paths always contain at least one inactive round (and this is reflected by the fact that these cases were usually the hardest to handle in our proof). Thus, a natural strategy would be to choose a key schedule that ensures that an inactive round necessarily costs many active Sboxes in the other (active) rounds. Eventually, we believe this direction will lead to a good understanding on how a key schedule should be designed when utilized for an AES-like primitive.

Acknowledgements

The authors would like to thank the anonymous referees for their helpful comments. Also, the third author would like to thank Sareh Emami, Jérémy Jean and Yang Li for fruitful discussions at ASK 2012 that initiated this work. The third and fourth authors are supported by the Singapore National Research Foundation Fellowship 2012 (NRF-NRFF2012-06) and Temasek Labs (DSOCL16194).

References

[BK09] Alex Biryukov and Dmitry Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 1–18. Springer, 2009.

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.

[BKN09] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. Distinguisher and Related-Key Attack on the Full AES-256 (Extended Version). *IACR Cryptology ePrint Archive*, 2009:241, 2009.

[BN10] Alex Biryukov and Ivica Nikolic. Automatic Search for Related-Key Differential Characteristics in Byte-Oriented Block Ciphers: Application to AES, Camellia, Khazad and Others. In Henri Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 322–344. Springer, 2010.

[BN11] Alex Biryukov and Ivica Nikolic. Search for Related-Key Differential Characteristics in DES-Like Ciphers. In Antoine Joux, editor, *Fast Software Encryption - 18th International Workshop, FSE 2011*, volume 6733 of *LNCS*, pages 18–34. Springer, 2011.

[BR11] Paulo S. L. M. Barreto and Vincent Rijmen. Whirlpool. In Henk C. A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security, 2nd Ed.*, pages 1384–1385. Springer, 2011.

[CZK+11] Jiali Choy, Aileen Zhang, Khoongming Khoo, Matt Henricksen, and Axel Poschmann. AES Variants Secure against Related-Key Differential and Boomerang Attacks. In Claudio Agostino Ardagna and Jianying Zhou, editors, *Information Security Theory and Practice. Security and Privacy of Mobile Devices in Wireless Communication - 5th IFIP WG 11.2 International Workshop, WISTP 2011*, volume 6633 of *LNCS*, pages 191–207. Springer, 2011.

[DF16] Patrick Derbez and Pierre-Alain Fouque. Automatic search of meet-in-the-middle and impossible differential attacks. In *CRYPTO*, pages 157–184. Springer, 2016.

[DFJ13] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. 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*, pages 371–387, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

[FJP13] Pierre-Alain Fouque, Jérémy Jean, and Thomas Peyrin. Structural Evaluation of AES and Chosen-Key Distinguisher of 9-Round AES-128. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013*, volume 8042 of *LNCS*, pages 183–203. Springer, 2013.

[GPPR11] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. In Preneel and Takagi [PT11], pages 326–341.

[JNP14a] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. `Deoxys` v1.1, 2014. Submission to the CAESAR competition, `http://www1.spms.ntu.edu.sg/~syllab/Deoxys`.

[JNP14b] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. `Joltik` v1.1, 2014. Submission to the CAESAR competition, http://www1.spms.ntu.edu.sg/~syllab/Joltik.

[JNP14c] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. `Kiasu` v1.1, 2014. Submission to the CAESAR competition, http://www1.spms.ntu.edu.sg/~syllab/Kiasu.

[JNP14d] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In Sarkar and Iwata [SI14], pages 274–288.

[Mat94] Mitsuru Matsui. On Correlation Between the Order of S-boxes and the Strength of DES. In Alfredo De Santis, editor, *Advances in Cryptology - EUROCRYPT '94*, volume 950 of *LNCS*, pages 366–375. Springer, 1994.

[MHM+02] Lauren May, Matthew Henricksen, William Millan, Gary Carter, and Ed Dawson. Strengthening the Key Schedule of the AES. In Lynn Margaret Batten and Jennifer Seberry, editors, *Information Security and Privacy - ACISP 2002*, volume 2384 of *LNCS*, pages 226–240. Springer, 2002.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology - Inscrypt 2011*, volume 7537 of *LNCS*, pages 57–76. Springer, 2011.

[Nik10] Ivica Nikolic. Tweaking AES. In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography - SAC 2010*, volume 6544 of *LNCS*, pages 198–210. Springer, 2010.

[PT11] Bart Preneel and Tsuyoshi Takagi, editors. *Cryptographic Hardware and Embedded Systems - CHES 2011*, volume 6917 of *LNCS*. Springer, 2011.

[SHW+14] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. In Sarkar and Iwata [SI14], pages 158–178.

[SI14] Palash Sarkar and Tetsu Iwata, editors. *Advances in Cryptology - ASIACRYPT 2014*, volume 8874 of *LNCS*. Springer, 2014.

[SIH+11] Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An Ultra-Lightweight Blockcipher. In Preneel and Takagi [PT11], pages 342–357.