Revisiting AES Related-Key Differential Attacks with Constraint Programming
David Gérault, Pascal Lafourcade, Marine Minier, Christine Solnon

To cite this version:
David Gérault, Pascal Lafourcade, Marine Minier, Christine Solnon. Revisiting AES Related-Key Differential Attacks with Constraint Programming. Information Processing Letters, Elsevier, 2018, 139, pp.24-29. 10.1016/j.ipl.2018.07.001. hal-01827727

HAL Id: hal-01827727
https://hal.archives-ouvertes.fr/hal-01827727
Submitted on 2 Jul 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Revisiting AES Related-Key Differential Attacks with Constraint Programming

David Gérault¹, Pascal Lafourcade¹, Marine Minierb,∗, Christine Solnonc

¹Université Clermont Auvergne, LIMOS, UMR 6158, F-63173, France
bUniversité de Lorraine, LORIA, UMR 7503, F-54506, France
cUniversité de Lyon, INSA-Lyon, F-69621, France, LIRIS, CNRS UMR5205

Abstract
The Advanced Encryption Standard (AES) is one of the most studied symmetric encryption schemes. During the last years, several attacks have been discovered in different adversarial models. In this paper, we focus on related-key differential attacks, where the adversary may introduce differences in plaintext pairs and also in keys. We show that Constraint Programming (CP) can be used to model these attacks, and that it allows us to efficiently find all optimal related-key differential characteristics for AES-128, AES-192 and AES-256. In particular, we improve the best related-key differential for the whole AES-256 and give the best related-key differential on 10 rounds of AES-192, which is the differential trail with the longest path. Those results allow us to improve existing related-key distinguishers, basic related-key attacks and \( q \)-multicollisions on AES-256.

Keywords: AES, Related-key attacks, Constraint Programming

Introduction

As attacking the Advanced Encryption Standard (AES) in the unknown key model seems to be out of reach at this time, many recent results focus on the so-called related-key, known-key or chosen-key models [1][2][3]. During the last decade, many results bring some grist to this research direction. In particular, the notion of differential \( q \)-multicollisions was introduced in [4]. A differential \( q \)-multicollision for a cipher \( E_{K′}(\cdot) \) is defined by a non zero key difference \( \delta K \), a non zero plaintext difference \( \delta X \) and a set of \( q \) distinct pairs \((X^i, K^i) \) with \( i \in [1,q] \) such that all \( E_{K′}(X^i) \oplus E_{K′\oplus\delta K}(X^i \oplus \delta X) \) are equal. Constructing such a \( q \)-multicollision for an ideal \( n \)-bit block cipher has a time complexity of \( \mathcal{O}(q \cdot 2^{q/2}n) \). However, for AES-256 the number of required AES encryptions has been shown to be equal to \( q \cdot 2^{67} \) in [4].

Building such \( q \)-multicollisions requires finding optimal (in terms of probability) related-key differential characteristics. This challenging task was tackled for AES-128 with a graph
traversal approach in [3], and for AES-128, AES-192, and AES-256 with a depth-first search approach in [4]. However, the 4-round solution for AES-128 claimed to be optimal in [4, 3] has been shown to be sub-optimal in [5]. In [5], the authors used Constraint Programming (CP) to efficiently enumerate related-key differential characteristics on AES-128.

In this paper, we further investigate the interest of using CP for finding optimal related-key differential characteristics for AES-192 and AES-256 whereas [4] has only focused on AES-128. We give new optimal solutions found with our CP approach. Table 1 sums up our new results in different attack models.

In Section 1, we give a brief overview of how our CP models work. In Section 2, we show how to use the solutions found by our CP models to improve existing related-key differential attacks for AES-192 and AES-256.

### 1. CP models for finding AES related-key differential paths

Mounting related-key differential attacks requires finding a related-key differential characteristic [3, 1], i.e. a plaintext difference $\delta X = X \oplus X'$ and a key difference $\delta K = K \oplus K'$, such that $\delta X$ becomes $\delta X_r$ after $r$ rounds with a probability as high as possible. The AES operations ShiftRows ($SR$), MixColumns ($MC$), AddRoundKey ($ARK$) are linear, i.e., they propagate differences in a deterministic way (with probability 1). The only non-linear operation is SubBytes ($SB$) where the used S-box $S$ transforms a given difference into another one in a probabilistic way. Even if the most important part of the AES KeySchedule ($KS$) is linear, it also makes regular calls to the S-box $S$.

To find optimal related-key differential characteristics and as done in [9] and [3], we use a two-step solving process. Step 1 works with a boolean representation of differences: We denote $\Delta A$ the boolean representation of the byte difference $\delta A$ such that $\Delta A = 0 \iff \delta A = 0$ and $\Delta A = 1 \iff \delta A \in [1, 255]$. These boolean variables give difference positions. The goal of Step 1 is to find a Boolean solution that assigns values to Boolean variables such that the AES transformation rules are satisfied. During this first step, the SubBytes operation $SB$ is not considered. Indeed, it does not introduce nor remove differences.
Then, Step 2 uses these positions to determine difference values at the byte level, i.e., to find the actual value $\delta A \in [1, 255]$ for each boolean variable $\Delta A$ which is equal to 1. Note that some solutions at the boolean level (found during Step 1) cannot be transformed into solutions at the byte level during Step 2. These solutions are said to be byte inconsistent.

1.1. Basic CP Model for Step 1

A first CP model for Step 1 may be derived from the AES transformations in a rather straightforward way. We extend the model described in [5] for AES-128 to AES-192 and AES-256. A CP model is defined by a set of variables, such that each variable $x$ has a domain $D(x)$, and a set of constraints, i.e., relations that restrict the values that may be simultaneously assigned to the variables. For each differential byte $\delta B$, we define a Boolean variable $\Delta B$ whose domain is $D(\Delta B) = \{0, 1\}$: it is assigned to 0 if $\delta B = 0$, and to 1 otherwise.

The XOR constraint for ARK and KS. We first define a XOR constraint for ARK and KS. Let us consider three differential bytes $\delta A$, $\delta B$ and $\delta C$ such that $\delta A \oplus \delta B = \delta C$. If $\delta A = \delta B = 0$, then $\delta C = 0$. If ($\delta A = 0$ and $\delta B \neq 0$) or ($\delta A \neq 0$ and $\delta B = 0$) then $\delta C \neq 0$. However, if $\delta A \neq 0$ and $\delta B \neq 0$, then we cannot know if $\delta C$ is equal to 0 or not: This depends on whether $\delta A = \delta B$ or not. When abstracting differential bytes $\delta A$, $\delta B$ and $\delta C$ with Boolean variables $\Delta A$, $\Delta B$ and $\Delta C$ (which only model the fact that there is a difference or not), we obtain the following definition of the XOR constraint: $\text{XOR}(\Delta A, \Delta B, \Delta C) \iff \Delta A + \Delta B + \Delta C \neq 1$.

Both ARK and KS are directly modeled with XOR constraints. The definition of KS depends on the key length. The CP model defined in [5] only considers 128 bit key length. We have extended it to 192 and 256 bit lengths in a rather straightforward way.

ShiftRows and MixColumns. SR simply shifts variables. The MDS property of $MC$ is ensured by posting a constraint on the sum of all variables on a same column before and after $MC$, which must belong to the set $\{0, 5, 6, 7, 8\}$.

Objective function. The goal is to minimize the number of S-boxes that must be crossed by Boolean differential paths. This is done in CP by introducing an integer variable $\text{obj}_{\text{Step1}}$ which is constrained to be equal to the sum of all boolean variables associated with bytes on which an non linear transformation $S$ is applied.

Limitations of the basic CP model. This basic CP model $CP_{\text{basic}}$ is complete, i.e., for any solution at the byte level (on $\delta$ variables), there exists a solution at the Boolean level (on $\Delta$ variables). However, preliminary experiments reported in [5] have shown us that there is a huge number of Boolean solutions which are byte inconsistent. For example, when the number of rounds is $r = 4$ for AES-128, the optimal cost is $\text{obj}_{\text{Step1}} = 11$, and there are more than 90 millions of Boolean solutions with $\text{obj}_{\text{Step1}} = 11$. However, none of these solutions is byte-consistent. In this case, most of the Step 1 solving time is spent at generating useless Boolean solutions which are discarded in Step 2.
1.2. Additional Constraints for Step 1

We have introduced in [5] a second model called $CP_{EQ}$ for AES-128. This model removes many byte-inconsistent solutions by propagating equality constraints at the byte level. In this paper, we extend it to AES-192 and AES-256. A full version of this model could be found in [10] and in [11].

New equality variables. For each couple of differential bytes $(\delta A, \delta B)$, we introduce a Boolean equality variable $EQ_{\delta A, \delta B}$ which is equal to 1 if $\delta A = \delta B$, and to 0 otherwise. These variables are constrained to define an equivalence relation by adding a symmetry constraint ($EQ_{\delta A, \delta B} = EQ_{\delta B, \delta A}$) and a transitivity constraint (if $EQ_{\delta A, \delta B} = EQ_{\delta B, \delta C} = 1$ then $EQ_{\delta A, \delta C} = 1$). Also, $EQ$ variables are related to $\Delta$ variables by adding the constraints:

$$(EQ_{\delta A, \delta B} = 1) \Rightarrow (\Delta A = \Delta B) \text{ and } EQ_{\delta A, \delta B} + \Delta A + \Delta B \neq 0$$

Revisiting the XOR constraint. When defining the constraint $XOR(\Delta A, \Delta B, \Delta C)$, if $\Delta A = \Delta B = 1$, then we cannot know if $\Delta C$ is equal to 0 or 1. However, whenever $\Delta C = 0$ (resp. $\Delta C = 1$), we know for sure that the corresponding byte $\delta C$ is equal to 0 (resp. different from 0), meaning that the two bytes $\delta A$ and $\delta B$ are equal (resp. different), i.e., that $EQ_{\delta A, \delta B} = 1$ (resp. $EQ_{\delta A, \delta B} = 0$). The same reasoning may be done for $\Delta A$ and $\Delta B$ because $(\delta A \oplus \delta B = \delta C) \Leftrightarrow (\delta B \oplus \delta C = \delta A) \Leftrightarrow (\delta A \oplus \delta C = \delta B)$. Therefore, we redefine the XOR constraint as follows:

$$XOR(\Delta A, \Delta B, \Delta C) \Leftrightarrow ((\Delta A + \Delta B + \Delta C \neq 1) \land (EQ_{\delta A, \delta B} = 1 - \Delta C) \land (EQ_{\delta A, \delta C} = 1 - \Delta B) \land (EQ_{\delta B, \delta C} = 1 - \Delta A))$$

Propagation of MDS at Byte Level. The MDS property ensures that, for each column, the total number of bytes which are different from 0, before and after applying $MC$, is either equal to 0 or strictly greater than 4. This property also holds for any xor difference between two different columns at different rounds. To propagate this property, for each pair of columns, we add a constraint on the sum of equality variables between bytes of these columns.

Constraints derived from KS. The KeySchedule mainly performs xor and S-box operations. As a consequence, each subkey byte $\delta K_i[j][k]$ at round $i$ may be expressed as a xor between bytes of the original key difference $\delta K[j][k]$, and bytes that have passed through an S-box at round $i - 1$, denoted by $\delta S(K_{i-1}[j][k])$. Hence, for each byte $\delta K_i[j][k]$, we precompute the set $V(i, j, k)$ such that $V(i, j, k)$ only contains bytes of $\delta K$ and $\delta S(K_{i-1})$ and $\delta K_i[j][k] = \bigoplus_{\delta A \in V(i, j, k)} \delta A$. For each set $V(i, j, k)$, we introduce a set variable $V_1(i, j, k)$ which is constrained to contain the subset of $V(i, j, k)$ corresponding to the Boolean variables equal to 1. We use these set variables to infer that two differential key bytes that have the same $V_1$ set are equal. Also, if $V_1(i, j, k)$ is empty (resp. contains one or two elements), we infer that $\Delta K_i[j][k]$ is equal to 0 (resp. a variable, or a xor between 2 variables).

1.3. CP Model for Step 2

Given a Boolean solution for Step 1, Step 2 aims at searching for the byte-consistent solution with the highest probability (or proving that there is no byte-consistent solution). Hence, for each Boolean variable $\Delta A$ of Step 1, we define an integer variable $\delta A$ whose domain depends on the value of $\Delta A$: If $\Delta A = 0$, then $D(\delta A) = \{0\}$ (i.e., $\delta A$ is also assigned to 0); otherwise, $D(\delta A) = [1, 255]$. As we look for a byte-consistent solution with maximal probability, we also add an integer variable $P_A$ for each byte $A$ that passes through an S-box: This variable corresponds to the base 2 logarithm of the probability $\Pr(\delta A \rightarrow \delta S_A)$ of obtaining the output difference $\delta S_A$ when the input difference is $\delta A$. If $\Delta A = 0$, then $\Pr(0 \rightarrow 0) = 1$ and therefore $D(P_A) = \{0\}$; otherwise, $\Pr(\delta A \rightarrow \delta S_A) \in \{2^{-256}, 4^{-256}\}$ and $D(P_A) = \{-7, -6\}$.

At byte level, the SubBytes transformation, which has no effect at the Boolean level, must be modeled. This is done thanks to a ternary table constraint which extensively lists all triples $(A, S_A, P_A)$ such that there exist two bytes $B_1$ and $B_2$ whose differences before and after passing through the S-box $S$ are equal to $A$ and $S_A$, respectively with a log2 probability equal to $P_A$. To find a byte-consistent solution with maximal differential probability, we maximize the sum of all $P_A$ variables.

2. From related-key differentials to related-key attacks

In this Section, we summarize the new AES related-key differential paths that we have computed with the new CP models previously described, and give new basic related-key attacks, related-key distinguishers and $q$-multicollisions that we are able to mount by using them for AES-192 and AES-256.

2.1. AES-192

Summary of related-key differential paths computed with CP. Using our CP approach, we found that the best related-key differential trail is on 10 rounds with 29 active S-boxes and a highest probability equal to $2^{-176}$: $2^{-37}$ coming from the keys and $2^{-139}$ from the ciphering part. The best differential characteristic is given in Table A.3 of Appendix A. We also give another trail with 30 active S-boxes where the differential characteristic has a probability equal to $2^{-188}$: $2^{-80}$ coming from the keys and $2^{-108}$ from the ciphering part. This trail is optimal for the probability in the state and is given in Table A.4 of Appendix A.

The first differential characteristic, which has an optimal probability, allows us to mount a basic related-key differential attack as done in [4]. The second one allows us to build a related-key distinguisher as it minimizes the probability in the state. We also provide in Table A.5 of Appendix A the best differential characteristic on 9 rounds of AES-192. This characteristic has a probability of $2^{-146}$ with 24 active S-boxes and is better than the one presented in [12].

Related-key distinguisher on 10 rounds. For this distinguisher, we use the related key differential characteristic given in Table A.4 in Appendix A which has a probability equal to $2^{-188}$. Considering the related-key distinguisher model, the probability that the differences correctly propagate through the internal states is $2^{-108} = 2^{-18 \cdot 6}$ as we have 18 active S-boxes in the
internal states, all with probability $2^{-6}$. It works for 1 out of $2^{80} = 2^{64} \cdot 2^{7.8}$ keys as we have 12 additional active S-boxes in the key schedule: 8 with probability $2^{-7}$ and the 4 others with probability $2^{-6}$. Therefore, the related-key distinguisher works with complexity $2^{108}$ for 1 out of $2^{80}$ related-key pairs on average.

**Basic related-key differential attack on 10 rounds.** We first change the trail given in Table A.3 of Appendix A in the deciphering direction to get more active S-boxes in the two last rounds to recover the key bytes implied at the input of those S-boxes. The new trail has eight active S-boxes in the last round on two anti-diagonals and two active S-boxes in the penultimate round.

Thus, we must find the 10 key bytes $K_{10}^*[0][2]$, $K_{10}^*[1][3]$, $K_{10}^*[2][0]$, $K_{10}^*[3][1]$, $K_{10}^*[0][3]$, $K_{10}^*[1][0]$, $K_{10}^*[2][1]$, $K_{10}^*[3][2]$, $K_{9}^*[0][2]$ and $K_{9}^*[0][3]$.

We use the following procedure from the ciphertexts for each of the $2^{37} \times 2^7$ key pairs:

1. Repeat $2^{47}$ times:
   (a) Compose two structures of $2^{64}$ ciphertexts with all possible values for the first and second anti-diagonals. Decrypt the first structure with $K$ and the second one with $K'$.
   (b) Sort the plaintexts and check for a pair with the correct input difference. Save these valid pairs if any.

2. For each of these pairs, derive $2^{16}$ variants for the 10 key bytes. There are 10 S-boxes in the last two rounds for which we know the input and output differences. Therefore, there are $2^{10} \cdot 8 \cdot 8 = 2^{16}$ possibilities for the 80 key bits per candidate pair without false alarms.

3. Pick the key candidate with the best occurrence.

The overall complexity of the whole procedure, which is repeated $2^{44}$ times, is $2^{47} + 65 = 2^{112}$ in data and time and $2^{65}$ in memory. We need to repeat $2^{47}$ times step 1.(a) and 1.(b) to keep on average $2^{47} \cdot 2^{44-109} = 2^4 = 16$ right pairs and $2^{47}$ wrong pairs and to completely discard false alarms. This gives us 80 key bits.

### 2.2. AES-256

**Summary of related-key differentials computed with CP.** The optimal byte solution for 14 rounds of AES-256 has a probability of $2^{-146}$, and it is given in Table A.2 of Appendix A. Note that the one given in [4] has a probability of $2^{-154}$ to happen. We also obtain 43 solutions with a probability of $2^{-147}$. We experimentally checked that the 7 bottom rounds of the AES conform to the expected probability by producing the wanted difference after $2^{30}$ pairs on average, as predicted by the trail.

---

2 where * stands for the classical InvMixColumns transformation applied on the keys.

3 the $2^7$ term comes from all the possible unknown differences at the S-box output generated through the key schedule for the possible values of $\delta K_{10}^*[0][2]$, $\delta K_{10}^*[1][3]$, $\delta K_{10}^*[0][3]$ and $\delta K_{10}^*[3][2]$. 

---
$q$-multicollisions. Using this optimized byte consistent differential solution, we are able to improve the attacks proposed in [4]. More precisely, the cost to compute a $q$-multicollision given in [4] depends on 11 active S-boxes with probability $2^{-67}$. With the new differential characteristic, we gain a factor 2 on this complexity leading to a time complexity equal to $q \cdot 2^{66}$ encryptions. In the same way, the time complexity to find partial $q$-multicollisions becomes $q \cdot 2^{36}$ instead of $q \cdot 2^{37}$.

Related-key distinguisher. Considering the related-key distinguisher, the probability that the differences correctly propagate through the internal states is $2^{-19.6} = 2^{-114}$ and it works for 1 out of $2^{32} = 2^{14} \cdot 2^{18}$ keys as we have 5 additional active S-boxes: 2 with probability $2^{-7}$ and 3 with probability $2^{-6}$. Hence, the related-key distinguisher has a data/time complexity equal to $2^{140} = 2^{114} \cdot 2^{26}$.

Related-key attack. The related-key attack described in [4] may be directly applied using our new differential characteristic. The procedure is about to be the same than the one of [4] and for the basic related-key attack given for 10 AES-192 rounds except that the two modified rounds are rounds 0 and 1. In this case, the number of key bits to test in the two first rounds is equal to 80 and for each of the $2^{32}$ possible key pairs, we need to repeat $2^{28}$ times the process with two structures of $2^{64}$ plaintexts. Thus after Step 1, we have on average 4 right pairs, and for each pair we derive $2^{16}$ possible values for the ten key bytes without false alarms. Thus, the overall complexity of this attack becomes $2^{125}$ in data and time while testing $2^{32}$ keys. The required memory is $2^{65}$.

References


Appendix A. Our New Related-Key Differential Paths for AES-192 and AES-256

<table>
<thead>
<tr>
<th>Round</th>
<th>$\delta X_i = X_i \oplus X'_i$</th>
<th>$\delta K_i = K_i \oplus K'_i$</th>
<th>$Pr(States)$</th>
<th>$Pr(Key)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>init.</td>
<td>addadb76 addadb76 addadb76 addadb76</td>
<td>addadb76 addadb76 addadb76 addadb76</td>
<td>$2^{-8}$</td>
<td>–</td>
</tr>
<tr>
<td>1</td>
<td>69000000 00000000 69000000 00000000</td>
<td>b59a9ab5 00000000 00000000 b59a9ab5</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>2</td>
<td>69000000 00000000 69000000 00000000</td>
<td>c4dbdb76 69000000 00000000 c4dbdb76</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>3</td>
<td>09000000 9a000000 00000000 9a000000</td>
<td>b59a9ab5 b59a9ab5 00000000 00000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>4</td>
<td>69000000 00000000 69000000 00000000</td>
<td>c4dbdb76 addadb76 00000000 00000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>5</td>
<td>09000000 00000000 00000000 00000000</td>
<td>b59a9ab5 00000000 00000000 b59a9ab5</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>6</td>
<td>69000000 00000000 69000000 00000000</td>
<td>c4dbdb76 69000000 00000000 69000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>7</td>
<td>00000000 00000000 00000000 00000000</td>
<td>2f9a9ab5 2f9a9ab5 2f9a9ab5 2f9a9ab5</td>
<td>$2^{-14}$</td>
<td>$2^{-14} \times 2^{-14}$</td>
</tr>
<tr>
<td>8</td>
<td>69000000 00000000 69000000 00000000</td>
<td>69000000 00000000 69000000 00000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>9</td>
<td>00000000 00000000 00000000 00000000</td>
<td>2f9a9ab5 00000000 00000000 2f9a9ab5</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>10</td>
<td>69000000 00000000 69000000 00000000</td>
<td>69000000 00000000 69000000 00000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>11</td>
<td>00000000 00000000 00000000 00000000</td>
<td>2f9a9ab5 2f9a9ab5 00000000 00000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>12</td>
<td>69000000 00000000 69000000 00000000</td>
<td>69000000 00000000 69000000 00000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>13</td>
<td>00000000 00000000 00000000 00000000</td>
<td>2f9a9ab5 00000000 00000000 2f9a9ab5</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
<tr>
<td>End/14</td>
<td>69000000 69000000 69000000 69000000</td>
<td>69000000 69000000 69000000 69000000</td>
<td>$2^{-14}$</td>
<td>–</td>
</tr>
</tbody>
</table>

Table A.2: Our own related-key differential on 14 AES-256 rounds that happens with a probability $2^{-146}$. 


Table A.3: Our first related-key differential on 10 AES-192 rounds that happens with a probability $2^{-176}$.

Table A.4: Our related-key differential on 10 AES-192 rounds with 30 active S-boxes and a probability of $2^{-188}$.

Table A.5: Our related-key differential on 9 AES-192 rounds with 24 active S-boxes and a probability of $2^{-146}$.