A Splice-and-Cut Cryptanalysis of the AES

Dmitry Khovratovich[1] and Christian Rechberger[2]

[1]Microsoft Research Redmond
[2]ENS Paris and Chaire France Telecom

May 28, 2011

Abstract

Since Rijndael was chosen as the Advanced Encryption Standard, improving upon 7-round attacks on the 128-bit key variant or upon 8-round attacks on the 256-bit key variant has been one of the most difficult challenges in the cryptanalysis of block ciphers for more than a decade. In this paper we present a novel technique of block cipher cryptanalysis with bicliques, which leads to the following results:

- The first key recovery attack on 9 out of 14 rounds of AES-256 with computational complexity $2^{253.1}$ and success rate 1.
- The first key recovery attacks on 8 out of 10 rounds of AES-128. The best attack has computational complexity $2^{124.8}$ and success rate 0.63.
- The first combination of a non-random property and an algorithm that allows to distinguish the *full 10-round* AES-128 from an ideal cipher in a non-trivial way. This may be interpreted as a weak deviation from an ideal behavior in a model where the adversary is allowed to choose the key. In addition of being the first evidence of non-randomness in AES-128, this has some relevance when AES-128 is used in a compression function of a cryptographic hash function.

In contrast to most shortcut attacks on AES variants, we *do not need any related-keys.* As our attacks are of high complexity, yet practically verified to large extent, they do not threaten the practical use of AES-128 or AES-256 in any way.
Keywords: Advanced Encryption Standard, AES, block cipher, hash function, meet-in-the-middle attack, splice-and-cut, key recovery, distinguisher, non-randomness

1 Introduction

The last ten years saw some progress in the cryptanalysis of block ciphers. However, the block cipher standard AES is almost as secure as it was 10 years ago in the strongest and the most practical model with a single unknown key. The former standard DES has not seen a major improvement since Matsui's seminal paper in 1993 [44].

In contrast, the area of hash function cryptanalysis is growing quickly, encouraged by the cryptanalysis MD5 [54], of SHA-0 [7,18] and SHA-1 [53], followed by a practical attack on protocols using MD5 [51,52], preimage attacks on Tiger [34] and MD5 [50], etc. As differential cryptanalysis [8], a technique originally developed for ciphers, was carried over to hash function analysis, cryptanalysts are now looking for the opposite: a hash analysis method that would give new results on block ciphers. So far the most well-known attempt is the analysis of AES with local collisions [9–12], but it is only applicable to the related-key model. In the latter model an attacker works with plaintexts and ciphertexts that are produced under not only the unknown key, but also under other keys, which are related to the first one in the way chosen by the adversary. Such a strong requirement is rarely practical and, thus, has not been consider to be a threat for the use of the AES. Also, there is no evidence that the local collision approach can facilitate an attack in the more practical and relevant single-key model.

AES and recent attacks. The Advanced Encryption Standard (AES) [21] is a 128-bit block cipher and one of the most widely used ciphers in the world. It has been an object for intensive cryptanalytic efforts since its design as Rijndael in 1998. The versions with 192- and 256-bit key have been subject to attacks in the related-key setting [10, 11] on the full cipher (12 and 14 rounds, respectively). In contrast, for AES-128 the best attacks are only on a variant reduced to 7 out of 10 rounds [28], do not progress in the number of rounds since 2000, and don't seem to be able to take any significant advantage of related keys [12]. The single-key attacks on AES-192 and AES-256 have not progressed much further, with non-marginal attacks on the 8-round versions appeared in 2010 [28] and 2007 [56], resp.. As a result, AES is widely recognized as the most difficult challenge in symmetric cryptanalysis.

Among the most popular attacks on block ciphers: differential [8], linear [44], impossible differential [6, 39], integral [41] — the first two were made obsolete by the design of AES. The predecessor of the integral attack, the square attack [20] could break up to 6 rounds of AES, it's extension to 7 rounds of AES-128 is extremely marginal [30]. The impossible differential method led to the first non-marginal attacks on 7 rounds of AES-128 [4] and 8 rounds of AES-256 [56]. However, the length of the impossible differential in AES is restricted [47], so the extension of the attack to more rounds is very difficult. Integral attacks faced the same problem, since the regular integral property is short and extends by one round only with multisets [28].

Meet-in-the-middle attacks on block ciphers did get less attention (exceptions are [16, 27, 29, 37, 55]), although they are probably the most practical in terms of data complexity. A simple meet-in-the-middle attack requires only a single plaintext/ciphertext pair. The limited use of these attacks must be attributed to the requirement for large parts of the cipher to be independent of particular key bits. For the ciphers with nonlinear key schedule, like AES and most AES candidates, this requirement is apparently strong. As a result, the number of rounds broken with this technique is rather small [27], which seems to prevent it from producing results on yet unbroken 8-, 9-, and 10-round (full) AES. We also mention that the collision attacks [24, 25] use some elements of the meet-in-the-middle framework.

Meet-in-the-middle attacks with bicliques. In this paper we demonstrate that the meet-in-the-middle attacks on block ciphers have great potential if enhanced by a new concept we call *bicliques*. The new approach originates from the so-called splice-and-cut framework [2, 3, 34] in the hash function cryptanalysis, led to the best preimage attacks on the SHA family of hash functions [1, 34], including the surprising attack on 43 (out of 64) rounds of SHA-256, which in terms of attacked number of rounds exceeds the capabilities of collision attacks [36, 48]. The crucial element of the recent splice-and-cut framework is the concept of the initial structure, which handles parts of the primitive which are not independent of any key bits, and, thus, may overcome the problems produced by the key schedule. A formal treatment of the properties exposed by initial structures has led us to the concept of bicliques with subsequent applications to block cipher cryptanalysis.

The paper is structured as follows. First, we describe the concept of bicliques and its application to block cipher cryptanalysis (Section 2). We demonstrate that this concept converges to a differential cryptanalysis framework, hence numerous tools like message modification, neutral bits, and rebound-like techniques (primarily known from the hash function analysis) are applicable. Then we present the first key-recovery attacks on 9-round AES-256 (Section 3) and 8-round AES-128 (Section 4). Section 6 is devoted to the analysis of AES-128 in a weaker model via a comparison to an ideal cipher. With the same technique, now in a variant more closely resembling the splice-and-cut framework, we show that a non-trivial nonrandom property can be constructed for the full AES-128 faster than for an ideal cipher. This is the first result on the full AES-128. Additional details on the attacks and the important practical verifications are provided in Section 7 and the Appendix. An extensive discussion section concludes the paper.

AES-128 secret key recovery					
rounds	data	workload/succ.rate	memory	method	reference
7	$2^{127.997}$	2^{120}	2^{64}	Square	[30], 2000
7	2^{32}	$2^{128-\epsilon}$	2^{100}	Square-collision	[32], 2000
7	$2^{117.5}$	2^{123}	2^{109}	Impossible	[4], 2007
7	$2^{112.2}$	$2^{117.2}$MA	2^{109}	Impossible	[42] 2008
7	2^{80}	$2^{113}+2^{123}$ precomp.	2^{122}	MitM	[25], 2009
7	$2^{106.2}$	$2^{117.2}$MA	$2^{94.2}$	Impossible	[43], 2010
7	2^{103}	2^{116}	2^{116}	Square-multiset	[28], 2010
8	$2^{126.33}$	$2^{125.42}$	2^{102}	Biclique	**New**
8	2^{127}	$2^{126.1}$	2^{32}	Biclique	**New**
AES-128 known/chosen key distinguisher					
7	-	2^{56}	-	Square	[40], 2007
7	-	2^{24} generic: 2^{64}	2^{16}	Rebound	[45], 2009
8	-	2^{48} generic: 2^{64}	2^{32}	Rebound	[33], 2010
7	-	2^{120} generic: 2^{128}	2^{8}	Splice&Cut	[49], 2011
10 (full)	-	$2^{126.3}$ generic: 2^{128}	2^{32}	Biclique	**New**
AES-256 secret key recovery					
8	$2^{127.997}$	2^{204}	2^{1044}	Square	[30], 2000
8	$2^{116.5}$	$2^{247.5}$	2^{45}	Impossible	[56], 2007
8	$2^{89.1}$	$2^{229.7}$MA	2^{97}	Impossible	[42] 2008
8	2^{37}	2^{206}	2^{203}	MitM	[24], 2008
8	2^{80}	2^{241}	2^{123}	MitM	[25], 2009
8	2^{113}	2^{196}	2^{129}	Square-multiset	[28], 2010
9	2^{120}	$2^{253.1}$	2^{8}	Biclique	**New**

Table 1: Overview of key-recovery attacks on AES-128 (7 or more rounds) and AES-256 (8 or more rounds) in *single-key* models, both secret and known/chosen. As success rates are often not given in the related literature we assume them to be 1.

2 Bicliques for key-recovery attacks on block ciphers

2.1 Concept of the biclique

The idea of a biclique can add a significant number of rounds to the meet-in-the-middle attack on block ciphers. The regular meet-in-the-middle attack splits the primitive into two parts with independent inputs. In turn, we split the cipher into three parts and require a relaxed condition of independency from the first two parts only. More precisely:

$$E: \quad P \underset{\mathcal{E}_1}{\longrightarrow} V \underset{\mathcal{E}_2}{\longrightarrow} S \underset{\mathcal{E}_3}{\longrightarrow} C,$$

and there exists an internal variable $v \in V$ that can be computed as follows:

$$P \xrightarrow[\mathcal{E}_1]{K^b \text{ not used}} v \xleftarrow[\mathcal{E}_2]{K^f \text{ not used}} S, \tag{1}$$

where K^f and K^b are independent parts of the key material. The rest of the key is denoted by K^g.

For the sub-cipher $\mathcal{E}_3$ we guess K^g and construct a specific structure of inputs $\{S_j\}$ and outputs $\{C_i\}$ such that

$$\forall i, j \quad S_j \xrightarrow[\mathcal{E}_3]{K^f=i, K^b=j} C_i. \tag{2}$$

The structure admits a representation as a bipartite graph where states are vertices and computations are edges. Since it is a complete bipartite graph, we call such a set *a biclique*.

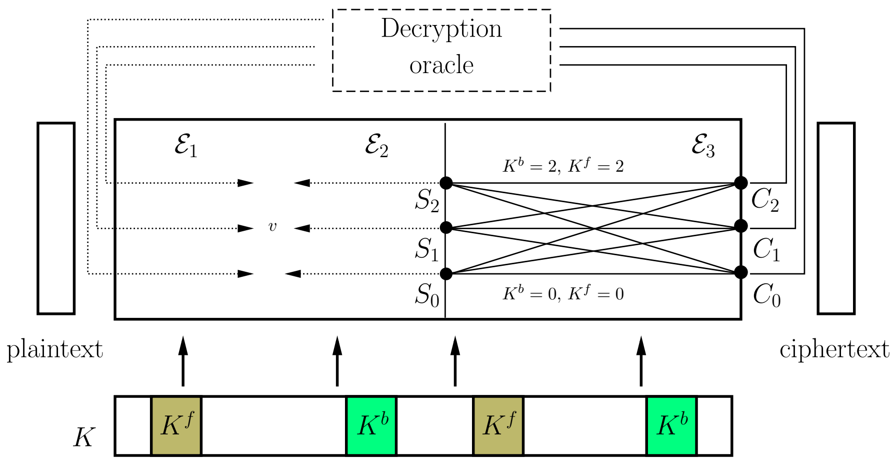

Figure 1: Biclique attack with three states and three ciphertexts.

Let us denote by P_i the plaintexts obtained from the decryption of $\{C_i\}$ under the secret key K_S. Suppose we have guessed K^g correctly, and $K_S = K^g \,||\, K^f = i || \, K^b = j$ for some i, j. Then

$$P_i \xrightarrow[\mathcal{E}_2 \circ \mathcal{E}_1]{K_S} S_j \xrightarrow[\mathcal{E}_3]{K_S} C_i \quad \Longrightarrow \quad P_i \xrightarrow[\mathcal{E}_2 \circ \mathcal{E}_1]{K^f = i || \, K^b = j} S_j,$$

which is detected by matching at v:

$$P_i \xrightarrow[\mathcal{E}_1]{K^f = i} v \xleftarrow[\mathcal{E}_2]{K^b = j} S_j.$$

For AES and other ciphers with byte-oriented key schedule it makes sense to consider a group of keys $\{K_{i,j}\}$ after a guess of K^g rather than to consider key bits:

$$K_{i,j}: \ K^f = i, \ K^b = j.$$

Therefore, the algorithm of the attack is outlined as follows (Figure 1):

1. Guess K^g or, equivalently, fix a group of keys $\{K_{i,j}\}$;
2. Prepare a biclique with states $\{S_j\}$, ciphertexts $\{C_i\}$, and keys $\{K_{i,j}\}$.
3. Ask for the decryption of ciphertexts C_i and get a set of plaintexts $\{P_i\}$;
4. Compute $P_i \xrightarrow{K_{i,*}} v_i$ and $v'_j \xleftarrow{K_{*,j}} S_j$ and check if $v_i = v'_j$ for some i, j. A match proposes a candidate key.
5. Re-check candidate keys on other variables in V and get the right key.

In the attacks, generally, $|\{S_j\}| = |\{C_i\}| = 2^d$ for some d. We call d the *dimension* of the biclique. Since the number of produced ciphertexts is proportional to the number of bicliques, we have to avoid covering the full codebook when keys are larger than the ciphertexts (like in AES-256). The approach that we propose is to keep only those bicliques whose ciphertexts belong to a particular set of cardinality smaller than the codebook.

2.2 Differential-based biclique attack

We use bicliques of dimension 1 in our attacks on AES since larger bicliques are very difficult to find for such a large number of rounds. Bicliques of dimension 1 are better constructed in terms of differentials and differential trails with the procedure resembling the rebound attack [46]. We are also able to amortize the construction cost of a biclique by producing many more out of a single one. The attack algorithm is outlined as follows (Figure 2).

1. Fix a key group $\{K_{0,0}, K_{0,1}, K_{1,0}, K_{1,1}\}$.

2. Construct a biclique:

   (a) Choose an intermediate state T in $\mathcal{E}_3$ and construct truncated differential trails from both S and C to T.

   (b) Inbound phase: Guess the differences in the differential trails up to T.

   (c) Get the values of T that satisfy the input and output differences.

   (d) Outbound phase: Use the remaining degrees in freedom in the state to sustain difference propagation in trails.

   (e) Output the states for the biclique. For longer keys some bicliques are filtered out.

3-5. Get plaintexts and check for the match in the middle.

6. Produce new bicliques out of the current one for other key groups.

In our attacks some steps are optimized. E.g., we guess not all differences in the trail, but only part of them, and subsequently filter out the solutions. We also denote the trails as follows: those originating from S are Δ-trails, and those originating from C are ∇-trails.

In the attack on AES-128, instead of fixing the key group, we fix only the difference between keys and derive actual values during the attack. The disadvantage of this approach is that key groups are generated online, and we have to take care of possible repetitions. We stress here that whenever we speak of differences between keys, these describe differences between the group of keys that are guessed. We never need access to decryptions under keys that are related by those differences, i.e. the attacks we describe are always in the single-key model.

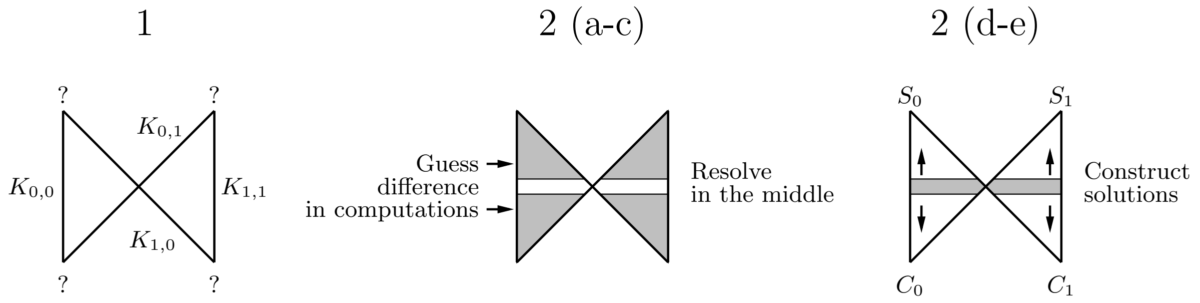

Figure 2: Differential-based biclique attack. We guess difference between computations and derive states S_j and ciphertext C_i as conforming elements.

3 Key-recovery attack on 9-round AES-256

3.1 Description of AES-256

AES-256 [21] is a block cipher with 128-bit plaintext and 256-bit key, whose internal state S and key K are treated as a 4×4 and 4×8 byte matrices $S_{i,j}$ and $K_{i,j}$. First, the key is xored to the plaintext and the resulting state is transformed by a sequence of 14 rounds. They process the state with the bytewise transformation SubBytes (simply SB), the byte permutation ShiftRows (SR), the linear transformation MixColumns (MC) which operates columnwise, and the subkey addition AddRoundKey (AK). The last rounds omits the MixColumns transformation.

SB is a nonlinear transformation operating on 8-bit S-boxes with maximum differential probability as low as 2^{-6} (for most cases 0 or 2^{-7}). The ShiftRows rotates bytes in row r by r positions to the left. The MixColumns is a linear transformation with branch number 5, i.e. in the column equation $(y_0, y_1, y_2, y_3) = MC(x_0, x_1, x_2, x_3)$ only 5 and more variables can be non-zero. The key schedule expands the key K into 8 256-bit expanded keys $K[0], K[1], \dots, K[7]$, which produce 15 128-bit subkeys $K^0, K^1, \dots, K^{14}$. The other details of the AES specification are irrelevant to our attack and we refer the reader to [21].

We enumerate the rounds from 1 to 14, and the internal state after round r by $S^{\text{AK-}r}$. The internal states inside round r are denoted as follows:

$$S^{\text{AK-}(r-1)} \xrightarrow{\text{SubBytes}} S^{\text{SB-}r} \xrightarrow{\text{ShiftRows}} S^{\text{SR-}r} \xrightarrow{\text{MixColumns}} S^{\text{MC-}r} \xrightarrow[K^r]{\text{AddRoundKey}} S^{\text{AK-}r},$$

The expanded keys $K[r]$ have the following property: one byte in a subkey $K[i]$ affects only two bytes in the subkey $K[i-1]$.

We also use the fact that the equation on the S-box input $S(x \oplus \delta) \oplus S(x) = \delta'$ has either 0, or 2, or 4 solutions in x for $\delta \neq 0$. In all cases the set of solutions form a linear space [23]. The same holds for the output of the S-box.

3.2 Attack

Our attack is differential-based biclique attack (Section 2.2).

Step 1.

A biclique of dimension 1 involves two states, two ciphertexts, and a group of four keys. The keys in the group are defined via the difference in subkeys, i.e. like in a related-subkey boomerang attack:

$$\begin{aligned}
&K_{0,1}: &&K_{0,1}[3] \oplus K_{0,0}[3] = \Delta K;\\
&K_{1,0}: &&K_{1,0}[3] \oplus K_{0,0}[3] = \nabla K;\\
&K_{1,1}: &&K_{1,1}[3] \oplus K_{0,1}[3] = \Delta K \oplus \nabla K.
\end{aligned}$$

The differences ΔK and ∇K are defined columnwise:

$$\Delta K = (\bar{0}, \bar{0}, \bar{0}, \bar{0}, A, A, A, A); \quad \nabla K = (B, B, \bar{0}, \bar{0}, \bar{0}, \bar{0}, \bar{0}, \bar{0}),$$

where

$$A = \text{MixColumns}\begin{pmatrix}0\\0\\2\\0\end{pmatrix}; \qquad B = \begin{pmatrix}0\\2\\\text{b9}\\2\end{pmatrix} = \text{MixColumns}\begin{pmatrix}*\\*\\0\\0\end{pmatrix}.$$

Let us note that the key relation in the next expanded key is still linear:

$$K_{1,0}[4] \oplus K_{0,0}[4] = K_{1,1}[4] \oplus K_{0,1}[4] = (B, \bar{0}, \bar{0}, \bar{0}, \bar{0}, \bar{0}, \bar{0}, \bar{0})$$

Evidently, the groups do not intersect and cover the full key space.

We split the 9-round AES-256 as follows:

- $\mathcal{E}_1$ is round 1.
- $\mathcal{E}_2$ is rounds 2-4.
- $\mathcal{E}_3$ is rounds 5-9.

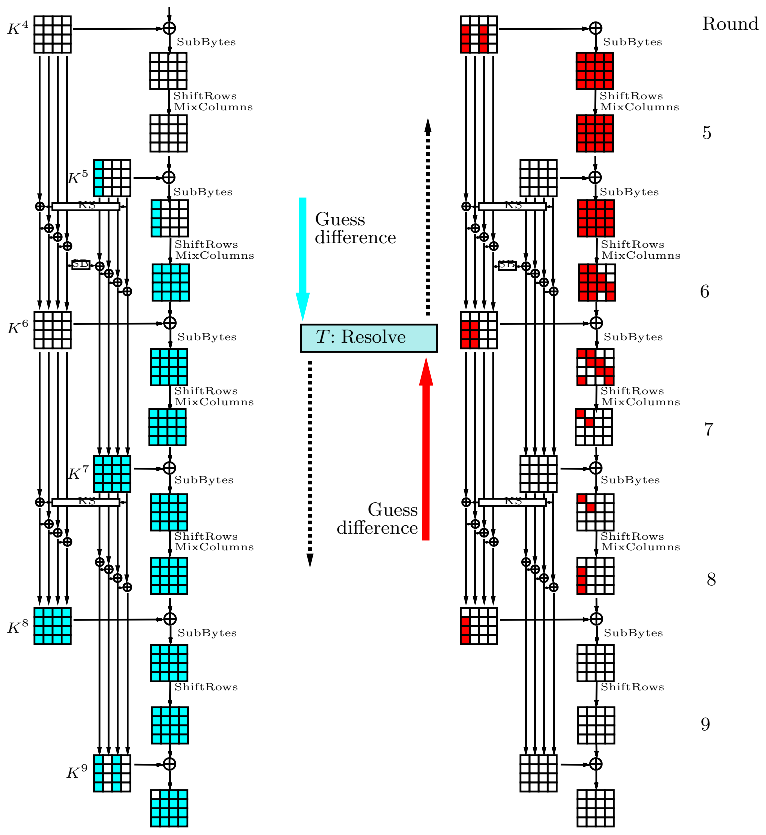

Figure 3: Biclique construction in AES-256. Δ-trail (left) and ∇-trail (right).

Step 2.

An illustration of steps 2(a) - 2(e) is given in Fig. 3.

Step 2 (a).

The intermediate state T in $\mathcal{E}_3$ is the S-box layer in round 7. We construct truncated differential trails in rounds 5-6 based on the injection of ΔK after round 5 (Figure 3, left), and in rounds 7-9 based on the injection of ∇K before round 9 (Figure 3, right).

Step 2 (b).

We guess the differences in the truncated trails up to T. We have four active S-boxes in round 6 and two active S-boxes in round 8. We also require Δ-trails be equal. In total we make $2^{7\cdot(4+2\cdot 2)} = 2^{56}$ guesses.

Step 2 (c).

For each S-box in round 7 that is active in both trails (eight in total) we take a quartet of values that conform to the input and output differences, being essentially the boomerang quartet for the S-box (one solution per S-box on average). For the remaining 8 S-boxes we take all possible values. Therefore, we have 2^{64} solutions for each guess in the inbound phase, or 2^{120} solutions in total.

Step 2 (d).

Outbound phase: we filter out the solutions that do not conform to the differential trails in rounds 6 and 8. We have four active S-boxes in each Δ-trail, and two active S-boxes in each ∇-trail, hence 12 in total. Therefore, we get a 84-bit filter, and leave with 2^{36} bicliques.

Step 2 (e).

Now we keep only the bicliques with byte $C_{0,0}$ equal to zero in both ciphertexts. This is a 16-bit filter, which reduces the number of bicliques to 2^{20}. We need only one.

Step 3-5.

We ask for the decryption of two ciphertexts and get two plaintexts. The matching position (v) is the byte $S^2_{0,0}$. As demonstrated in Fig. 4, it is equal as a function of the plaintext for keys with difference ΔK (not affected by lightblue cells), and is also equal as a function of S for keys with difference ∇K (not affected by red cells). We compute v in both directions and check for the match.

Step 6.

We can produce sufficiently many bicliques out of one to amortize the construction cost. Let us look at the subkey K^6 in the outbound phase. We can change its value to any of the 2^{96} specific values so that the active S-boxes in round 6 during the outbound phase are not affected. On the other hand, any change in bytes in rows 1,2,3 affects only those rows in the subkeys K^8 and K^9 and hence does not affect $C_{0,0}$. Therefore, we have $128 - 32 - 32 = 64$ neutral bits in K^6.

Similarly, we identify 9 bytes in K^7 that can be changed so that K^6, the active S-boxes in round 8, and the byte $C_{0,0}$ are unaffected. Those are bytes in the first three columns not on the main diagonal. Therefore, we have 72 neutral bits in K^7, and 136 neutral bits in total.

3.3 Complexity

A single biclique with the $C_{0,0} = 0$ is constructed with complexity $2^{120-20} = 2^{100}$ and 2^8 memory for table lookups at Step 2 (c). However, 136 neutral bits in the key reduce the amortized construction cost significantly. Let us compute the cost of constructing a new biclique according to Step 6. A change in a single byte in K^7 needs 5 S-boxes, 1 MixColumn and several XORs recomputing for each ciphertext, which gives us the complexity of 10/16 AES rounds. This change also affects two bytes of K^5, so we have to recompute one half of round 5, with the resulting complexity of 1 AES round per biclique. The total amortized complexity is 1.625 AES rounds.

In the matching part we compute a single byte in two directions, thus spending 9/16 of a round in rounds 1-3, and full round 4, i.e. 3.125 full rounds per biclique. In total we need 4.75 AES rounds per biclique, i.e. $2^{-0.92}$ 9-round AES-256 calls. The complexity generated by false positives is at most 2^{-6} rounds per biclique. We need 2^{254} bicliques, so the total complexity is $2^{253.1}$.

The data complexity is 2^{120} since one ciphertext byte is always fixed. The success rate of the attack is 1, since we can generate many bicliques for each key group.

4 Key-recovery attack on 8-round AES-128

4.1 Description of AES-128

AES-128 differs from AES-256 in the number of rounds (10) and the key schedule procedure that is omitted here. We only note that expanded keys are equivalent to subkeys: $K[r] = K^r$, so a flip in a single byte of subkey K^r changes only two bytes of subkey K^{r-1}.

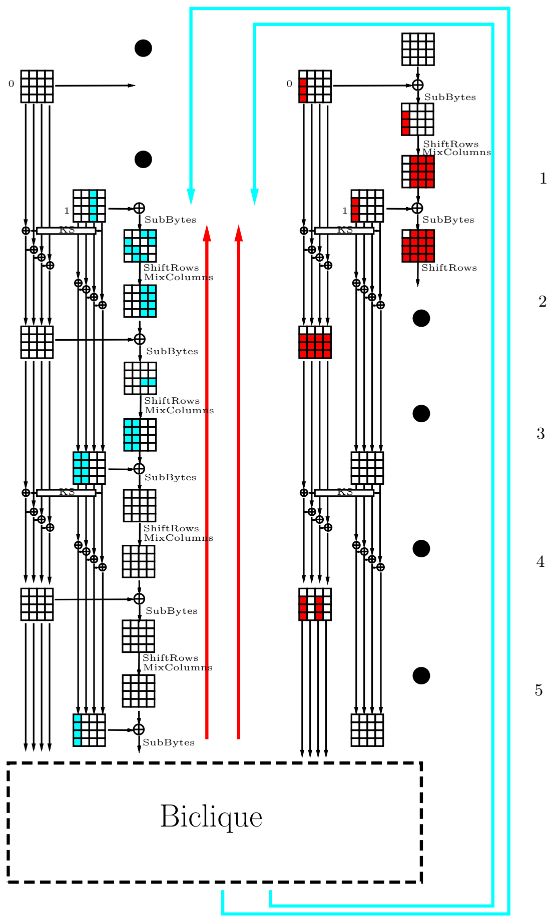

Figure 4: Matching in AES-256. Byte $S_{0,0}$ after round 1 can be computed in each direction.

We also view the sequence $SB \circ AK \circ MC \circ SR \circ SB$ as a layer of four parallel 32-bit *super boxes* [22], each parameterized with the corresponding column of the subkey. We decided to include SR to the Super Box for more clarity.

4.2 Attack

Our attack is differential-based biclique attack (Section 2.2).

Step 1.

A biclique of dimension 1 involves two states, two ciphertexts, and a group of four keys. The keys in the group are defined via the difference in subkeys, i.e. like in a related-subkey boomerang attack:

$$\begin{aligned} K_{0,1}: &\qquad K_{0,1}[4] \oplus K_{0,0}[4] = \Delta K; \\ K_{1,0}: &\qquad K_{1,0}[6] \oplus K_{0,0}[6] = \nabla K; \\ K_{1,1}: &\qquad K_{1,1}[4] \oplus K_{0,1}[4] = \Delta K. \end{aligned}$$

The differences ΔK and ∇K are defined columnwise:

$$\Delta K = (A, \bar{0}, \bar{0}, \bar{0}); \quad \nabla K = (B, \bar{0}, B, \bar{0}),$$

where

$$A = \begin{pmatrix} 0 \\ * \\ * \\ * \end{pmatrix} = \text{MixColumns} \begin{pmatrix} 0 \\ 0 \\ * \\ * \end{pmatrix}; \qquad B = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Let us note that ∇K in round 8 (∇K^8) is equal to $(B, \bar{0}, \bar{0}, \bar{0})$.

Instead of fixing the full keys in the group, we fix only three key bytes in the last column of K^5 so that we know the key difference in the Δ-trail in round 6.

We split the 8-round AES-128 as follows:

- $\mathcal{E}_1$ is round 1.
- $\mathcal{E}_2$ is rounds 2-3.
- $\mathcal{E}_3$ is rounds 4-8.

Step 2.

An illustration of the biclique construction in steps 2(a) - 2(e) is given in Fig. 5.

Step 2 (a).

The intermediate state in $\mathcal{E}_3$ is the Super-Box layer in rounds 6-7. We construct truncated differential trails in rounds 5 based on the injection of ΔK after round 4 (Fig. 5, left), and in rounds 7-8 based on the injection of ∇K^8 after round 8 (Fig. 5, right). Given the keys in the group, we know the key differences in trails.

Step 2 (b).

We guess the actual differences in the truncated trails. We have three active S-boxes in round 5 and one active S-boxes in round 8. In total we make $2^{7 \cdot 4 \cdot 2} = 2^{56}$ guesses.

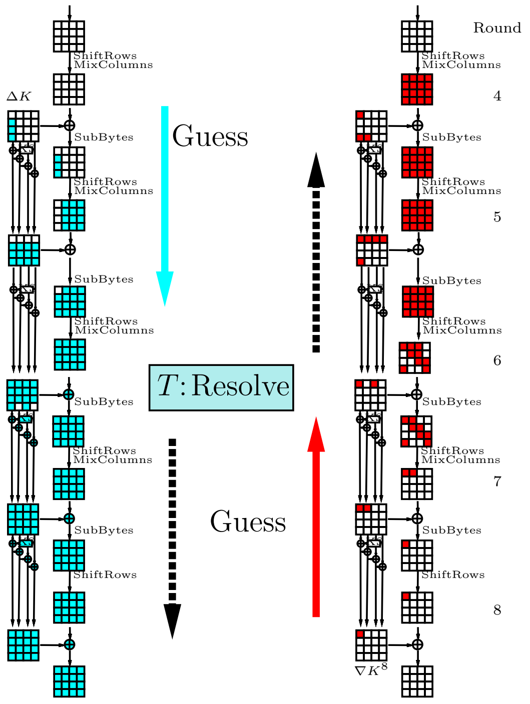

Figure 5: Biclique construction in AES-128.

Step 2 (c).

First, for Super-boxes in columns 0 and 1 we construct all possible solutions that conform to the input and output differences. We note that the K^6 bytes not adjacent to active S-boxes in the ∇-trail in round 7 do not affect the ∇-trail, and thus can be left undefined. Therefore, we construct $2^{64-32} = 2^{32}$ solutions. Similarly, we construct all possible solutions for Super-boxes in columns 2 and 3. We get only 2^{24} solutions since we have restricted three bytes of K^5.

Step 2 (d).

Outbound phase: we combine the solutions for pairs of Super-boxes and filter out those that are incompatible with the S-box behavior guessed in rounds 6 and 8. In round 8 we have a full 14-bit filter, and in round 6 we have only a portion of filter via the differences in the extended ∇-trails. For the latter, the filter is 6 bit per active S-box, with the remaining 8-bit filter to be fulfilled by adjusting the key. Therefore, we have $2^{32+24-14-6\cdot 3} = 2^{24}$ solutions. In those solutions we have fixed 64 bits of K^6 and 24 bits of K^5, which gives 80 bits in total due to dependencies. Finally, we additionally fix 24 key bits to sustain the difference propagation in round 6. Taking the guess of differences into account, we have constructed 2^{80} bicliques with 104 key bits fixed. The remaining $126 - 104 = 22$ key bits that define the key group can be chosen arbitrarily, so we amortize the construction of a biclique. As a result, we construct 2^{102} bicliques for each value of the three bytes of K^5 we have fixed in advance.

Step 2 (e).

We do not restrict the ciphertexts.

Step 3-5.

We ask for the decryption of two ciphertexts and get two plaintexts. The matching position (v) is the state $S^2_{2,3}$. We compute v in both directions and check for the match (Figure 6).

Step 6.

We construct 2^{22} bicliques out of one by choosing 22 bits of the key (defined in Step 2 (d)) so that difference propagation in the guessed parts remains untouched. The simplest change that does not affect the trails is the flip in two bytes of K^6 not adjacent to the active S-boxes and simultaneously in four bytes of K^5 so that three active S-boxes in round 5 are stable.

4.3 Complexity

Solutions for the Super-boxes are constructed online by substituting 2^{32} input pairs to the super-box transformation and filtering out incorrect quartets. This gives the time complexity 2^{32} and the memory complexity 2^{16}. Therefore, Step 2 (c) has complexity 2^{32}. It also dominates the complexity of other steps, so we construct 2^{80} bicliques with 104 fixed key bits in $2^{56+32} = 2^{88}$, and 2^{102} bicliques in 2^{88} plus the time required to construct a new biclique (Step 6). The latter we estimate as recomputation of 1/2 of round 5, full round 4, 1/8 of round 7, 1/2 of round 8 per biclique ciphertext, or 4.25 AES rounds per biclique. In the matching phase we recompute 9/16 of a round in rounds 1-3, i.e. 1.125 rounds per biclique.

The chance of getting a false positive is $4 \cdot 2^{-8} = 2^{-6}$ per biclique. Most of false positives require only round 2 to recompute, which gives 2^{-9} AES calls overhead on average, which is negligible compared to the matching phase. Therefore, the total complexity of the attack with 2^{126} bicliques is about

$$2^{126}\left(4.25/8 + 2^{-9} + 1.125/8\right) = 2^{125.42}.$$

with 2^{32} memory and 2^{127} data.

4.4 Success rate

Our attack always outputs the right key as soon as it belongs to one of the quartets produced in the attack. As the key bits are adaptively chosen in the attack, the algorithm does not guarantee that the quartets are pairwise different. On the other hand, each quartet has equal chance to be produced. Therefore, we estimate that the algorithm generates a natural proportion of $(1 - 1/e) = 63\%$ quartets. If we keep track of quartets in the loop after the guess of three bytes of K^5, then the memory complexity grows to 2^{102}. For a success probability of 63% the second variant of the attack produces $2^{125.33}$ bicliques in $2^{124.75}$ time, and needs $2^{126.33}$ chosen ciphertexts. The workload/success rate ratio is thus $2^{125.42}$, which is arguably impossible for any generic method.

5 Splice-and-cut framework for producing distinguishing properties of block ciphers in the single-chosen-key model

5.1 The model for the distinguishing property

Simply being able to claim that AES-128 is not an ideal cipher does not carry any cryptanalytic meaning, as Canetti, Goldreich, and Halevi noted already in [17] that any cipher (or function ensemble) with a compact description can be easily distinguished from an ideal cipher (or equivalently from a random oracle [19, 35]).

The model we are going to propose does more than that, it uses a particular non-trivial property that is provably hard to exhibit when facing an ideal cipher, yet for the first time (slightly) easier

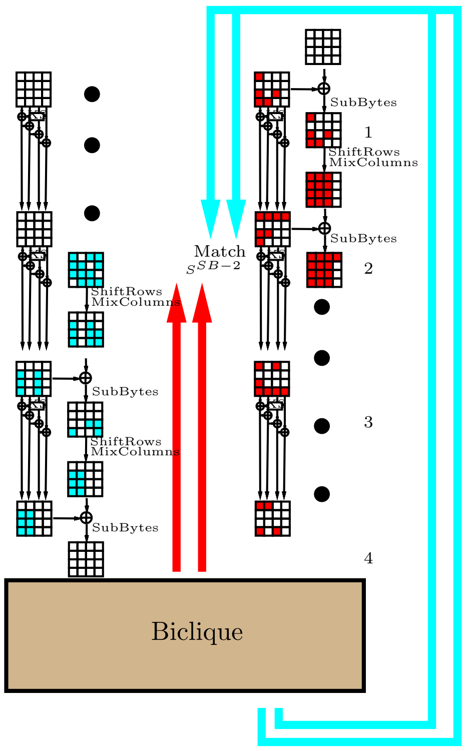

Figure 6: Matching in the 8-round attack on AES-128.

for AES-128. We use the following chosen-key scenario: On one hand, we have the cryptanalyst with access to the description and hence all the internals of AES-128. On the other hand, we have a computationally unbounded entity we call Adam that has only black-box access to an ideal cipher and it's inverse. Adam gets to see the following interface: Input: a 128-bit text x, a 128-bit key k, and a single-bit b to choose between the cipher and it's inverse. Output: if b indicates the inverse, it will output $E_k^{-1}(x)$, otherwise it will output $E_k(x)$.

For both the cryptanalyst and the entity Adam the challenge is as follows. Given a particular T, find k and x such that $E_k(x) + x = T$. For every query Adam is making the cryptanalyst is allowed to perform computations that are equivalent to the computation of a single AES computation. Black et al. [14, 15] derive a lower and an upper bound for the number of queries to an ideal cipher for finding a preimage for a compression function that uses this ideal cipher in e.g. Davis-Meyer mode. For this mode the lower bound states that q queries to the ideal cipher produces a preimage with probability *at most* $q \cdot 2^{n-1}$. Therefore, a preimage algorithm should be considered an attack as long as a success rate 2^{-s} is reached with complexity lower than 2^{n-s-1}.

5.2 Bicliques for preimages

We apply the idea of the biclique to the preimage attack as follows. Assume we attack a compression function based on a block cipher in the Davies-Meyer mode. Then the first and the last round of the cipher (now the compression function) can be considered as consecutive as long as we are given the hash value [2]. We find a preimage with a procedure similar to the meet-in-the-middle attacks on block ciphers. The main differences is that a plaintext and a ciphertext are now related bytewise. Therefore, we do not need to know the full ciphertext to bypass the feedforward. As a result, the biclique can be located in the middle of the primitive.

Let us denote the top state of the biclique in this approach by A, and the bottom state by B.

Then we get the following equations out of Equation (2)

$$\forall i, j \quad A_i \xrightarrow[\mathcal{E}_1]{M^b=i||M^f=j} B_j, \tag{3}$$

where M^b and M^f are neutral message bits not involved into the forward and the backward computation, respectively.

The attack procedure is roughly the same as in the key-recovery attack with the main difference that we do not have to track the key quartets, since we do not have to test all of them. Each biclique of dimension d produces a preimage with probability 2^{2d-n}. Let us express the success rate via d and n and get the following bound on the complexity T_S of each biclique computation and check:

$$T_S < 2^{2d-1}. \tag{4}$$

6 Distinguisher for the full AES-128

Now we describe a distinguisher for the full 10-round AES-128 in the following dense: we demonstrate that finding preimages for AES-128 in Davies-Meyer mode requires fewer operations than for an ideal cipher.

In contrast to AES-256, where a distinguisher has been found for the full cipher with 14 rounds, the progress of attacks on AES-128 in different modes is moderate. The first 7-round attack by Knudsen and Rijmen [40] was followed by rebound attacks on 7 [45] and 8 rounds [33], and a boomerang distinguisher on 8 rounds [13]. We also point out the recent work by Sasaki [49], which targets the same model as we do in this section, but attacks only a 7-round version.

Our attack is a splice-and-cut preimage attack enhanced with bicliques. We generate bicliques of dimension 1 in rounds 5-7 and match in $S_{2,3}^{SB-2}$ (Figure 7). In the description of the attack we follow the algorithm used in single-key recovery attacks in previous sections.

Step 1

Since plaintext and ciphertext are now related bytewise, we can move the starting point further away from the ciphertext and exploit the absence of MixColumns in the last round similarly to the attack in [49]. We also significantly extend the matching part compared to the key-recovery attack.

The difference ΔK in the key group is chosen as follows and defines the difference between subkeys K^4 keys in the group of keys:

$$\Delta K = (A, 0, 0, 0), \text{ where } A = (0, 0, *, *)^T = \text{MixColumns}\,(0, *, *, *)^T.$$

The value of ∇K is chosen adaptively. We require that $\nabla K(K_{1,0}^3) = 0$, which is equivalent to

$$\nabla K(K_{2,0}^{10}) = K_{2,3}^3 \oplus K_{2,3}^7,$$

with the other bytes of ∇K equalling zero.

We fix $K_{2,3}^3$ and $K_{2,3}^7$ in advance and then derive an appropriate $\nabla K_{2,0}^{10}$. We stress that the active S-box in round 7 in the key schedule is not affected by the difference $\Delta K(K^7)$.

For convenience, we express $K_{2,3}^3$ and $K_{2,3}^7$ as functions of a subset of K^6. The smallest set of the K^6 bytes that determines those bytes is $\mathcal{K} = \{K_{1,2}^6, K_{1,3}^6, K_{2,0}^6, K_{2,1}^6, K_{2,2}^6, K_{2,3}^6, K_{3,2}^6, K_{3,3}^6\}$. Therefore, we define $\nabla K(K^{10})$ for every $\mathcal{K}$.

Step 2.

First, we construct truncated differential trails up to the Super-box layer in rounds 6-7. Then we guess input differences in the Δ-trail (differences in the ∇-trail are known). Then we construct solutions for the Super-box layer for every value of relevant K^6 bytes:

Column	0	1	2	3
Varying key bytes	$K^6_{0,0}, K^6_{1,0}, K^6_{3,0}$	$K^6_{0,1}, K^6_{1,1}, K^6_{3,1}$	$K^6_{0,2}$	$K^6_{0,3}$
Solutions	2^{24}	2^{24}	2^8	2^8

We combine solutions in columns 0 and 2 and produce 2^{32} combinations. The same holds for combinations in columns 1 and 3.

Now we filter out the 128-bit combinations that are incompatible with guesses. This filter is equivalent to a 28-bit filter on the values of active S-boxes. Therefore, we filter out incorrect combinations with complexity about 2^{36}, leaving with $2^{32+32-28} = 2^{36}$ full bicliques per guess.

Steps 3-5.

Then we match at $S^{SB-2}_{2,3}$. In the backward direction this byte is affected by the difference ΔK. However, we do have enough information for the partial matching, since the ΔK goes through only a single S-box. As a result, we can either compute this S-box two times, or generate a set of 2^7 impossible values, that can not be produced in $S^{SB-2}_{2,3}$.

Step 6

We generate 2^{16} bicliques out of one by applying the difference $\Delta^N = (B, 0, 0, 0)$ to K^7, where A is the column of the form

$$A = \text{MixColumns}(*, *, 0, 0)^T.$$

We apply Δ^N to K^7 and recompute the biclique in the forward direction — note that no active S-boxes in the ∇-trail are affected (Figure 7). In the backward direction, we apply the difference $(A, A, 0, 0)$ to K^6 and recompute the biclique from round 6 in backward direction — again, no active S-box in the Δ-trail is affected.

Complexity

In step 2 we generate 2^{36} bicliques with complexity 2^{36}. We use table lookups in the manner described in the 8-round attack, hence requiring 2^{32} additional computation and 2^{32} memory. The amortized cost is determined by Step 6 of the attack, where we produce 2^{16} bicliques out of one with complexity about 3 AES rounds per biclique. However, they cover only 2^8 different $\mathcal{K}$ and thus have many repetitions. In total, we generate 2^{64} bicliques for 2^8 sets $\mathcal{K}$. We repeat this procedure for $2^{62-8} = 2^{54}$ different $\mathcal{K}$ thus generating 2^{126} bicliques.

Both matching variants have complexity less than 1 AES round for the match itself. The computation of $S^{SB-2}_{2,3}$ requires full round 8, one half of rounds 9-10, one quarter of round 1, i.e. 2.25 rounds per state to compute. For the backward computation we compute full rounds 3-4 and an equivalent of full round 2, i.e. 3 rounds per state. In total we compute an equivalent of $1.35 = 2^{0.4}$ AES calls per biclique. The memory requirement is 2^{32}.

Let us substitute the attack parameters to Equation (4): $2^{0.4} < 2^{2-1}$, which is clearly true. As a result, our attack has an advantage of at least a factor of $2^{0.6}$ over any black-box attack, and arguably higher over AES-specific attacks. The time complexity of the attack with 2^{126} bicliques is $2^{126} \cdot 2^{0.4} = 2^{126.4}$ AES calls and has success probability close to 1.

7 On practical verification

Especially for the type of cryptanalysis described in this paper were carring out an attack in full is computationally infeasible, practical verification of attack details and steps is important in order to confidence in it. To address this, we explicitly state the following:

- We verified all truncated differentials through 8-round and 10-round AES-128 key-schedules, and through 9-round AES-256 key-schedule.
- We implemented the technically most complex part of our attack: the biclique construction (for the AES-256 attack). We verified the complexity estimate, and also give an example of a biclique in Table 2 in the Appendix.
- We checked the distribution of super-box output differences. We checked that it is random enough where we require randomness, though some non-random behavior was detected and might be important for constructing bicliques of high dimension over super-boxes.
- We verified that some difference guesses must be equal like in the AES-256 attack due to the branch number of MixColumns that results in the correlation of differences in the outbound phase.

8 Discussion and Conclusions

The new approach for AES cryptanalysis poses in no way a threat to its practical use. To conclude, we discuss the properties of AES-128 and AES-256 that allowed us to cover more rounds than done before, discuss what distinguishes our approach from a more optimized implementation of a brute-force attack, and list a number of problems to consider for future work.

8.1 What properties of the AES allowed to obtain these new results

Our approach heavily relies on key schedule properties, especially taking advantage of the relatively slow backwards diffusion. Whereas using key-schedule properties in related-key attacks is natural, there seem only a few examples in the literature where this is used in the arguably more relevant single-key setting. This includes the attack on the self-synchronized stream cipher Moustique [38], the lightweight block cipher KTANTAN [16], and recent improvements upon attacks on 8-round attacks on AES-192 and AES-256 [28].

Also, as already observed in [27, 49], the fact that the MixColumn transformation is omitted in the last round of AES helps to design attacks for more rounds.

8.2 An approach to distinguish between bruteforce-like and shortcut-like "attacks"

The fact that most results in this paper are relatively close to generic attacks gives rise to the following question. What makes them different from an implementation of a simple brute-force attack with improved efficiency? We'd like to address this in several ways:

- Speed-ups due to optimizations of brute force will also speed-up the shortcut methods we propose. Also, whenever it is difficult to be precise about certain parts of our estimates, we choose to be conservative, potentially resulting in an underestimate of the claimed improvement.
- The highest inner loop counter is below 2^n for a success probability for which a brute force attack would need 2^n repetitions.

- To the best of our knowledge, there are no generic methods known that would speed-up key-recovery attacks given a part of the codebook. Likewise, there are no known methods that would benefit from accessing some internal state (i.e. memory accesses) for exhibiting a non-random property in the chosen-key model.

Let us expand on the second point here. One may want to distinguish between approaches that repeat some inner loop 2^n times and approaches that need less repetitions to recover a full n-bit key. Examples of the former include the early approaches (from 2000) to attacks on 7-round AES by Ferguson et al. [30] and Gilbert and Minier [32], but also rather recent results on preimage attacks on the SHA-3 candidates Hamsi by Fuhr [31] and by Dinur and Shamir [26] and on reduced Keccak by Bernstein [5].

Note that the matching phase in our approach allows to extend the number of rounds in a relatively simple way. The cost however is that the added rounds have to be computed 2^n times, and hence would not qualify as a shortcut-like "attack" as discussed above.

8.3 Open Problems

There are a number of other settings this approach may be applied to. It will be interesting to study other block ciphers like the AES finalists or more recent proposals with respect to this class of attacks. Also, we may decide to drop the requirement of the biclique to be complete, i.e. instead of a complete bipartite graph consider a more general graph. There may be cases where different tradeoffs between success probability, complexity requirements, and even number of rounds are obtainable. Alternatively, this paper may inspire work on more generic attacks on block ciphers that try to take advantage of the fact that a part of the codebook is available.

Acknolwedgements

We thank Joan Daemen and Vincent Rijmen for their helpful feedback on earlier versions of this paper. Part of this work was done while Christian Rechberger was with KU Leuven and visiting MSR Redmond. This work was supported by the European Commission under contract ICT-2007-216646 (ECRYPT II).

References

[1] Kazumaro Aoki, Jian Guo, Krystian Matusiewicz, Yu Sasaki, and Lei Wang. Preimages for step-reduced SHA-2. In *ASIACRYPT'09*, volume 5912 of *Lecture Notes in Computer Science*, pages 578–597. Springer, 2009.

[2] Kazumaro Aoki and Yu Sasaki. Preimage attacks on one-block MD4, 63-step MD5 and more. In *Selected Areas in Cryptography'08*, volume 5381 of *Lecture Notes in Computer Science*, pages 103–119. Springer, 2008.

[3] Kazumaro Aoki and Yu Sasaki. Meet-in-the-middle preimage attacks against reduced SHA-0 and SHA-1. In *CRYPTO'09*, volume 5677 of *Lecture Notes in Computer Science*, pages 70–89. Springer, 2009.

[4] Behran Bahrak and Mohammad Reza Aref. A novel impossible differential cryptanalysis of AES. In *Proceedings of the Western European Workshop on Research in Cryptology 2007 (WEWoRC'07)*, pages 152–156, 2007.

[5] Daniel J. Bernstein. Second preimages for 6 (7? (8??)) rounds of Keccak? NIST mailing list, 2010.

[6] Eli Biham, Alex Biryukov, and Adi Shamir. Miss in the middle attacks on IDEA and Khufu. In *FSE'99*, volume 1636 of *Lecture Notes in Computer Science*, pages 124–138. Springer, 1999.

[7] Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. Collisions of SHA-0 and reduced SHA-1. In *EUROCRYPT'05*, volume 3494 of *Lecture Notes in Computer Science*, pages 36–57. Springer, 2005.

[8] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. *J. Cryptology*, 4(1):3–72, 1991.

[9] Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds. In *EUROCRYPT'10*, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer, 2010.

[10] Alex Biryukov and Dmitry Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In *ASIACRYPT'09*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2009.

[11] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolić. Distinguisher and related-key attack on the full AES-256. In *CRYPTO'09*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, 2009.

[12] 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 *EUROCRYPT'10*, volume 6110 of *Lecture Notes in Computer Science*, pages 322–344. Springer, 2010.

[13] Alex Biryukov and Ivica Nikolic. New Cryptanalysis of AES-128. Rump session of Crypto 2010, 2010.

[14] John Black, Phillip Rogaway, and Thomas Shrimpton. Black-box analysis of the block-cipher-based hash-function constructions from PGV. In *CRYPTO'02*, volume 2442 of *Lecture Notes in Computer Science*, pages 320–335. Springer, 2002.

[15] John Black, Phillip Rogaway, Thomas Shrimpton, and Martijn Stam. An Analysis of the Blockcipher-Based Hash Functions from PGV. *J. Cryptology*, 23(4):519–545, 2010.

[16] Andrey Bogdanov and Christian Rechberger. A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In *SAC'10*, volume 6544 of *Lecture Notes in Computer Science*, pages 229–240. Springer, 2010.

[17] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.

[18] Florent Chabaud and Antoine Joux. Differential collisions in SHA-0. In *CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 56–71. Springer, 1998.

[19] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. The Random Oracle Model and the Ideal Cipher Model Are Equivalent. In *CRYPTO'08*, volume 5157 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2008.

[20] Joan Daemen, Lars R. Knudsen, and Vincent Rijmen. The Block Cipher Square. In Eli Biham, editor, *FSE'97*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer, 1997.

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

[22] Joan Daemen and Vincent Rijmen. Understanding two-round differentials in AES. In Roberto De Prisco and Moti Yung, editors, *SCN'06*, volume 4116 of *Lecture Notes in Computer Science*, pages 78–94. Springer, 2006.

[23] Joan Daemen and Vincent Rijmen. Plateau characteristics and AES. *IET Information Security*, 1(1):11–17, March 2007.

[24] Hüseyin Demirci and Ali Aydin Selçuk. A meet-in-the-middle attack on 8-round AES. In *FSE'08*, volume 5086 of *Lecture Notes in Computer Science*, pages 116–126. Springer, 2008.

[25] Hüseyin Demirci, Ihsan Taskin, Mustafa Çoban, and Adnan Baysal. Improved Meet-in-the-Middle Attacks on AES. In *INDOCRYPT'09*, volume 5922 of *Lecture Notes in Computer Science*, pages 144–156. Springer, 2009.

[26] Itai Dinur and Adi Shamir. An Improved Algebraic Attack on Hamsi-256. Cryptology ePrint Archive, Report 2010/602, 2010. `http://eprint.iacr.org/`.

[27] Orr Dunkelman and Nathan Keller. The effects of the omission of last round's MixColumns on AES. *Inf. Process. Lett.*, 110(8-9):304–308, 2010.

[28] Orr Dunkelman, Nathan Keller, and Adi Shamir. Improved Single-Key Attacks on 8-Round AES-192 and AES-256. In *ASIACRYPT'10*, volume 6477 of *Lecture Notes in Computer Science*, pages 158–176. Springer, 2010.

[29] Orr Dunkelman, Gautham Sekar, and Bart Preneel. Improved meet-in-the-middle attacks on reduced-round DES. In *INDOCRYPT'07*, volume 4859 of *Lecture Notes in Computer Science*, pages 86–100. Springer, 2007.

[30] Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David Wagner, and Doug Whiting. Improved cryptanalysis of Rijndael. In *FSE'00*, volume 1978 of *Lecture Notes in Computer Science*, pages 213–230. Springer, 2000.

[31] Thomas Fuhr. Finding Second Preimages of Short Messages for Hamsi-256. In *ASIACRYPT'10*, volume 6477 of *Lecture Notes in Computer Science*, pages 20–37. Springer, 2010.

[32] Henri Gilbert and Marine Minier. A Collision Attack on 7 Rounds of Rijndael. In *AES Candidate Conference*, pages 230–241, 2000.

[33] Henri Gilbert and Thomas Peyrin. Super-Sbox cryptanalysis: Improved attacks for AES-like permutations. In *FSE'10*, volume 6147 of *Lecture Notes in Computer Science*, pages 365–383. Springer, 2010.

[34] Jian Guo, San Ling, Christian Rechberger, and Huaxiong Wang. Advanced Meet-in-the-Middle Preimage Attacks: First Results on Full Tiger, and Improved Results on MD4 and SHA-2. In *ASIACRYPT'10*, volume 6477 of *Lecture Notes in Computer Science*, pages 56–75. Springer, 2010.

[35] Thomas Holenstein, Robin Künzler, and Stefano Tessaro. Equivalence of the Random Oracle Model and the Ideal Cipher Model, Revisited. *To appear in STOC 2011.*

[36] Sebastiaan Indesteege, Florian Mendel, Bart Preneel, and Christian Rechberger. Collisions and Other Non-random Properties for Step-Reduced SHA-256. In *Selected Areas in Cryptography'08*, volume 5381 of *Lecture Notes in Computer Science*, pages 276–293. Springer, 2008.

[37] Takanori Isobe. A single-key attack on the full GOST block cipher. In *FSE'11 Preproceedings*, 2011.

[38] Emilia Käsper, Vincent Rijmen, Tor E. Bjørstad, Christian Rechberger, Matthew J. B. Robshaw, and Gautham Sekar. Correlated Keystreams in Moustique. In *AFRICACRYPT'08*, volume 5023 of *Lecture Notes in Computer Science*, pages 246–257. Springer, 2008.

[39] Lars R. Knudsen. DEAL - a 128-bit Block Cipher. Technical report, Department of Informatics, University of Bergen, Norway, 1998.

[40] Lars R. Knudsen and Vincent Rijmen. Known-key distinguishers for some block ciphers. In *ASIACRYPT'07*, volume 4833 of *Lecture Notes in Computer Science*, pages 315–324. Springer, 2007.

[41] Lars R. Knudsen and David Wagner. Integral cryptanalysis. In Joan Daemen and Vincent Rijmen, editors, *FSE'02*, volume 2365 of *Lecture Notes in Computer Science*, pages 112–127. Springer, 2002.

[42] Jiqiang Lu, Orr Dunkelman, Nathan Keller, and Jongsung Kim. New impossible differential attacks on AES. In *INDOCRYPT'08*, volume 5365 of *Lecture Notes in Computer Science*, pages 279–293. Springer, 2008.

[43] Hamid Mala, Mohammad Dakhilalian, Vincent Rijmen, and Mahmoud Modarres-Hashemi. Improved Impossible Differential Cryptanalysis of 7-Round AES-128. In *INDOCRYPT'10*, volume 6498 of *Lecture Notes in Computer Science*, pages 282–291. Springer, 2010.

[44] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In *EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1993.

[45] Florian Mendel, Thomas Peyrin, Christian Rechberger, and Martin Schläffer. Improved Cryptanalysis of the Reduced Grøstl Compression Function, ECHO Permutation and AES Block Cipher. In *Selected Areas in Cryptography'09*, volume 5867 of *Lecture Notes in Computer Science*, pages 16–35. Springer, 2009.

[46] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl. In *FSE'09*, volume 5665 of *Lecture Notes in Computer Science*, pages 260–276. Springer, 2009.

[47] Raphael C.W. Phan and M.U. Siddiqi. Generalised impossible differentials of Advanced Encryption Standard. *Electronics Letters*, 37(14):896–898, 2001.

[48] Somitra Kumar Sanadhya and Palash Sarkar. New collision attacks against up to 24-step SHA-2. In *INDOCRYPT'08*, volume 5365 of *Lecture Notes in Computer Science*, pages 91–103. Springer, 2008.

[49] Yu Sasaki. Meet-in-the-Middle Preimage Attacks on AES Hashing Modes and an Application to Whirlpool. In *FSE'11 Preproceedings*, 2011.

[50] Yu Sasaki and Kazumaro Aoki. Finding Preimages in Full MD5 Faster Than Exhaustive Search. In *EUROCRYPT'09*, volume 5479 of *Lecture Notes in Computer Science*, pages 134–152. Springer, 2009.

[51] Marc Stevens, Arjen K. Lenstra, and Benne de Weger. Chosen-prefix collisions for MD5 and colliding X.509 certificates for different identities. In *EUROCRYPT'07*, volume 4515 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 2007.

S_0			
40	8a	ba	52
30	4a	10	52
34	b6	84	52
b8	fe	aa	52

S_1			
44	d2	66	7b
32	34	6e	f7
36	f4	b0	7a
b8	ba	71	3a

C_0			
79	18	c0	8e
67	ac	89	9e
2e	39	52	84
3c	fd	40	26

C_1			
5d	08	b5	ac
e5	bd	d3	54
a0	ac	d9	8a
09	6a	55	1e

$K_{0,0}[3]$							
7d	8a	d8	a4	30	e8	0	0
12	a8	f9	31	5a	42	0	0
12	55	cd	0b	32	d6	0	0
58	66	d8	cf	54	f8	0	0

$K_{0,1}[3]$							
7d	8a	d8	a4	**34**	**ec**	**4**	**4**
12	a8	f9	31	**58**	**40**	**2**	**2**
12	55	cd	0b	**30**	**d4**	**2**	**2**
58	66	d8	cf	**52**	**fe**	**6**	**6**

$K_{1,0}[3]$							
7d	8a	d8	a4	30	e8	0	0
10	**aa**	f9	31	5a	42	0	0
ab	**ec**	cd	0b	32	d6	0	0
5a	**64**	d8	cf	54	f8	0	0

$K_{1,1}[3]$							
7d	8a	d8	a4	**34**	**ec**	**4**	**4**
10	**aa**	f9	31	**58**	**40**	**2**	**2**
ab	**ec**	cd	0b	**30**	**d4**	**2**	**2**
5a	**64**	d8	cf	**52**	**fe**	**6**	**6**

Table 2: Example of a biclique for AES-256. S_i are states after MixColumns in round 5, C_i are ciphertexts.

[52] Marc Stevens, Alexander Sotirov, Jacob Appelbaum, Arjen K. Lenstra, David Molnar, Dag Arne Osvik, and Benne de Weger. Short chosen-prefix collisions for MD5 and the creation of a rogue ca certificate. In *CRYPTO'09*, volume 5677 of *Lecture Notes in Computer Science*, pages 55–69. Springer, 2009.

[53] Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding collisions in the full SHA-1. In *CRYPTO'05*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005.

[54] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In *EUROCRYPT'05*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2005.

[55] Lei Wei, Christian Rechberger, Jian Guo, Hongjun Wu, Huaxiong Wang, and San Ling. Improved meet-in-the-middle cryptanalysis of KTANTAN. Cryptology ePrint Archive, Report 2011/201, 2011. `http://eprint.iacr.org/`.

[56] Wentao Zhang, Wenling Wu, and Dengguo Feng. New results on impossible differential cryptanalysis of reduced AES. In *ICISC'07*, volume 4817 of *Lecture Notes in Computer Science*, pages 239–250. Springer, 2007.

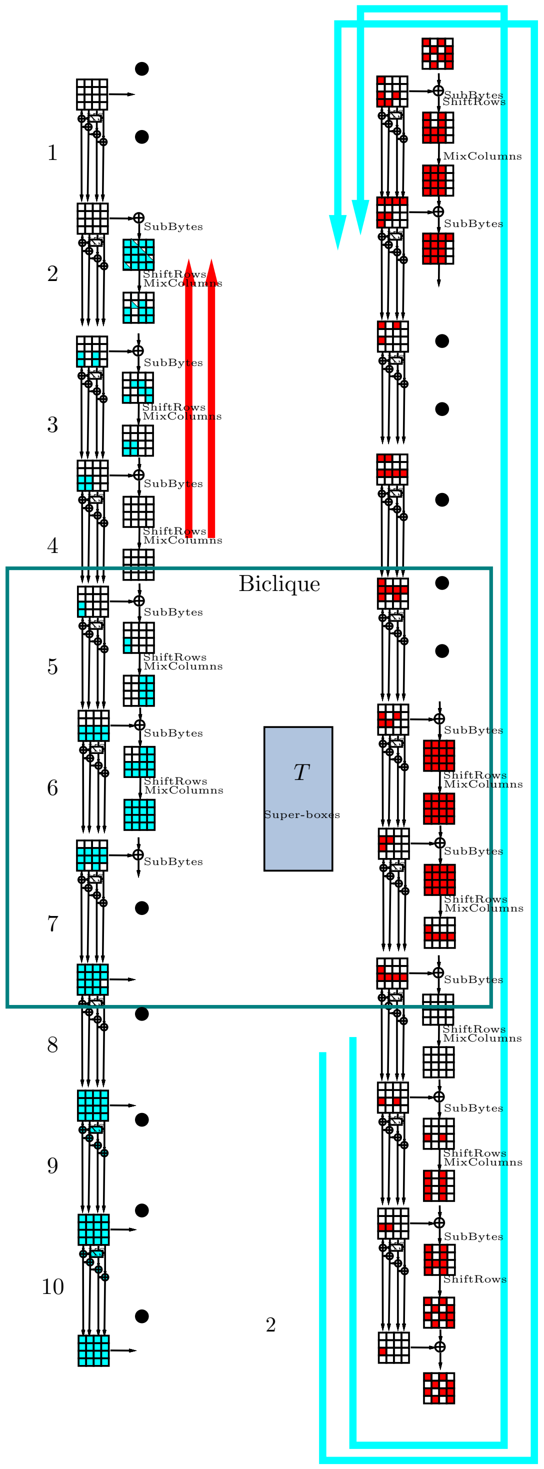

Figure 7: Layout of the 10-round distinguisher.