# Rectangle and Impossible-differential Cryptanalysis on Versions of ForkAES

Jannis Bossert, Eik List and Stefan Lucks

Bauhaus-Universität Weimar
firstname.lastname(at)uni-weimar.de

**Abstract.** The rapid distribution of lightweight devices raised the demand for efficient encryption and authenticated encryption schemes for small messages. For this purpose, Andreeva et al. recently proposed forkciphers, which fork the middle state within a cipher and encrypt it twice further under two smaller independent permutations. So, forkciphers can produce two output blocks which can allow to authenticate and encrypt small messages more efficiently.
As instance of particular interest, Andreeva et al. proposed ForkAES, a tweakable forkcipher based on the AES-128 round function, which forks the state after five out of ten rounds. While their authenticated encrypted schemes were accompanied by proofs, the security discussion for ForkAES could not be covered in their work, and founded on existing results on the AES and KIASU-BC; so, the study of advanced differential attacks remained to be filled by the community.
This work tries to foster the understanding of the security of ForkAES. It outlines a rectangle and an impossible-differential attack on nine rounds in the single-key related-tweak model; moreover, it describes a rectangle attack on ten rounds for a fraction of approximately $2^{32}$ keys. We emphasize that our results do not break ForkAES in the single-key setting, but shed more light on its security margin.

**Keywords:** AES · differential cryptanalysis · tweakable block cipher

## 1 Introduction

**Lightweight Designs.** The fast distribution of resource-constrained devices demands novel approaches for efficient encryption and authentication of short messages. The recent NIST call for submissions on lightweight cryptography emphasized the need of a standard [Nat18]. Though, since the analysis of new proposals takes considerable time, schemes and primitives based on well-analyzed operations are preferable.

**Forkciphers.** Forkciphers are a recent proposal by Andreeva et al. [ARVV18] for lightweight encryption of short messages. A (tweakable) fork cipher is similar to a (tweakable) block cipher: it takes a key and a plaintext block, and computes a ciphertext block. However, it also computes a second ciphertext block from the same input under an independent permutation. The crucial idea of their work is to speed up the computation by deriving the second output block from the state in the middle, and only reencrypt it over the remaining rounds twice. Therefore, forkciphers could yield efficient AE schemes since the computation of a ciphertext and tag reduces for messages of length at most a block.
As instance of particular interest, [ARVV18] proposed ForkAES, which employs the original key schedule and round function of the AES-128. Moreover, ForkAES is a tweakable block cipher that adopts the concept from KIASU-BC [JNP14]: in every round where the round key is XORed to the state, an additional 64-bit public tweak $T$ is XORed to the topmost two state rows. So, ForkAES encrypts the plaintext over the first five rounds exactly as in

**Table 1:** Summary of our related-tweak cryptanalysis of ForkAES. CP = #chosen plaintexts; Encs. = encryption equivalents; MAs = memory accesses.

| | | | | Time | | Data | Memory | |
|---|---|---|---|---|---|---|---|---|
| #Rds. | Model | \|Key space\| | Attack type | (Encs.) | (MAs) | (CP) | (bytes) | Ref. |
| 9 | Single-key | Full | Impossible Diff. | $2^{75.4}$ | $2^{110.2}$ | $2^{70.2}$ | $2^{104}$ | Sect. 4 |
| 9 | Single-key | Full | Rectangle | $2^{86.6}$ | $2^{92.4}$ | $2^{85}$ | $2^{90.4}$ | Sect. 3.2 |
| 10 | Weak-key | $\approx 2^{32}$ | Rectangle | $2^{100.6}$ | $2^{106.7}$ | $2^{99}$ | $2^{104.4}$ | Sect. 3.1 |

the KIASU-BC. Thereupon, it forks the state $X$ and produces from it a ciphertext $C_0$ exactly as KIASU-BC with the round keys $K^5$ through $K^{10}$, and a second ciphertext $C_1$ under six further round keys $K^{11}$ through $K^{16}$. As a minor difference to the AES, ForkAES does not omit the MixColumns operation in the final round. So, ForkAES requires only five additional rounds and six more key-schedule iterations to compute a second output block that can be used for authentication.

**Existing Security Arguments.** The adoption of the AES round function and the tweak process from KIASU-BC allow to profit from existing results. It is well-known that any four-round differential characteristic of the AES has at least 25 active S-boxes [DR02] in the single-key model. For KIASU-BC, however, the number decreases to eight since the tweak is also public, and provides additional degrees of freedom to the adversary [JNP14]. Andreeva et al. also considered meet-in-the-middle attacks briefly. Against other attacks, they state that: "the security of our forkcipher design can be reduced to the security of the AES and KIASU ciphers for further type of attacks" [ARVV18, Sect 3.2]. Concerning the computation of $C_0$, which is computed almost exactly as in KIASU-BC and only adds a final MixColumns operation, the designers' claim is clearly correct. However, the forking operation produces a difference between not only a pair of two states in the middle, but forms a quartet, which demands closer analysis. Thus, it is an interesting task for the community to study ForkAES in more detail.[1]

**Contribution.** This work describes a related-tweak rectangle attack on ten-round ForkAES for a subset of $2^{32}$ keys among the space. It further describes a rectangle attack in the single-key related-tweak setting and an impossible-differential attack on nine rounds. Their properties are summarized in Table 1. Our attacks are made possible by the forking step, which produces rectangle quartets from pairs of plaintexts.
We emphasize that our attacks do not endanger the security of the full ForkAES construction in the single-key setting nor that of the general forkcipher concept. However, our results contradict the designer's claim quoted above: impossible-differential attacks apply (at the moment) to at most seven rounds of AES-128, and to at most eight for KIASU-BC [DL17, TAY16]. Thus, the security margin of ForkAES is smaller than anticipated.
The research direction of lightweight AES-based designs in general and that of ForkAES and forkciphers in particular appear promising. Similar as for previous AES-round-based constructions (e.g. Simpira [GM16], or Haraka [KLMR16]), we can anticipate further developments of the construction. Throughout this work, we focus on the ForkAES construction from the original ePrint publication [ARVV18].

**Outline.** The remainder is structured as follows: after Section 2 briefly revisited the necessary preliminaries, Section 3 provides related-key rectangle attacks. Thereupon, Section 4 describes an impossible-differential attack. Section 5 concludes.

[1]We can anticipate other groups to also work on fostering the understanding of the security properties of ForkAES. We are open to all constructive comments or collaborations.

## 2 Preliminaries

### 2.1 Notations and Security Definitions

**General Notation.** We assume that the reader is familiar with the concepts of block ciphers and their cryptanalysis. Most of the time, we consider bit strings of fixed length. We mostly use uppercase letters (e.g., $X$) for bit strings, lowercase letters for indices ($x$), and calligraphic letters for sets ($\mathcal{X}$). Given some positive integer $n$, we interpret bit strings $X \in \{0,1\}^n$ as vector elements of $\mathbb{F}_2^n$, where addition is the bit-wise XOR, denoted by $\oplus$. Moreover, the AES works on byte vectors or byte matrices, i.e., 16-element vectors in $\mathbb{F}_{2^8}$. So, we interpret byte matrices of $r$ rows and $c$ columns as elements of $\mathbb{F}_{2^8}^{r \times c}$. We denote by $|X|$ the length in bits of a bit string $X = (X_{n-1}, \ldots, X_1, X_0)$; by $X \,\|\, Y$ the concatenation of two bit strings $X$ and $Y$; and by $X_i$ the $i$-th bit of $X$, where $X_0$ denotes the least and $X_{n-1}$ the most significant bit. We define by $\mathsf{lsb}_m(X) =^{\text{def}} (X_{m-1}, \ldots, X_0)$ and by $\mathsf{msb}_m(X) =^{\text{def}} (X_{n-1}, \ldots, X_{n-m})$. Given an integer $x$, $\langle x \rangle$ denotes the encoding of $x$ as an $n$-bit string, where the length $n$ should be clear from the context. Given sets $\mathcal{X}$ and $\mathcal{T}$, we define $\mathsf{Perm}(\mathcal{X})$ for the set of all permutations over $\mathcal{X}$, and $\widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{X}) = \{\widetilde{\pi} \mid \widetilde{\pi} : \mathcal{T} \times \mathcal{X} \to \mathcal{X}\}$ for the set of all tweakable permutations over $\mathcal{X}$ such that for each element $\widetilde{\pi} \in \widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{X})$, $\widetilde{\pi}^T(\cdot)$ is a permutation for all $T \in \mathcal{T}$. We write $X \twoheadleftarrow \mathcal{X}$ to denote that $X$ is sampled uniformly and independently at random from $\mathcal{X}$.
An adversary $\mathbf{A}$ is assumed to be an information-theoretic distinguisher that interacts with a set of oracles, and can ask queries to them. $\mathbf{A}$ has to distinguish between a real world $\mathcal{O}_{\text{real}}$ and an ideal world $\mathcal{O}_{\text{ideal}}$. The real world is given by a real construction keyed under some key $K \twoheadleftarrow \mathcal{K}$ sampled uniformly at random from some given key space $\mathcal{K}$. We write $\Delta_{\mathbf{A}}(\mathcal{O}_{\text{real}}; \mathcal{O}_{\text{ideal}}) =^{\text{def}} |\Pr[\mathbf{A}^{\mathcal{O}_{\text{real}}} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathcal{O}_{\text{ideal}}} \Rightarrow 1]|$ for the distinguishing advantage of $\mathbf{A}$. For a security notion $\mathsf{xx}$ on a cryptographic scheme $F$, We write $\mathbf{Adv}_F^{\mathsf{xx}}(\mathbf{A})$ for the maximal advantage of $\mathbf{A}$ to distinguish $F$ from the ideal primitive.

**Block Ciphers and Tweakable Block Ciphers.** Let $\mathcal{B} = \{0,1\}^n$ be some block space for some fixed integer $n$, hereafter. A TBC $\widetilde{E}$ with associated key space $\mathcal{K}$, tweak space $\mathcal{T}$, and message space $\mathcal{B}$ is a mapping $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{B} \to \mathcal{B}$ such that for every key $K \in \mathcal{K}$ and tweak $T \in \mathcal{T}$, it holds that $\widetilde{E}(K, T, \cdot)$ is a permutation over $\mathcal{B}$. We also write $\widetilde{E}_K^T(\cdot)$ as short form of $\widetilde{E}(K, T, \cdot)$ in the remainder.

**Definition 1** (TPRP Advantage)**.** Let $\mathcal{K}$ be a non-empty set and $\mathcal{B}$, and $\mathcal{T}$ be message and tweak space, respectively. Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{B} \to \mathcal{B}$ denote a tweakable block cipher. Let $\widetilde{\pi} \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{B})$ and $K \twoheadleftarrow \mathcal{K}$. Then, the TPRP advantage of an adversary $\mathbf{A}$ w.r.t. $\widetilde{E}$ is defined as $\mathbf{Adv}_{\widetilde{E}}^{\text{TPRP}}(\mathbf{A}) =^{\text{def}} \Delta_{\mathbf{A}}(\widetilde{E}_K; \widetilde{\pi})$.

**Forkciphers.** A tweakable forkcipher $\widetilde{E}$ is a tuple of three deterministic algorithms: An encryption algorithm $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{B} \to (\mathcal{B})^2$; a decryption algorithm $\widetilde{D} : \mathcal{K} \times \mathcal{T} \times \mathcal{B} \times \{0,1\} \to \mathcal{B}$; and a tag-reconstruction algorithm $\widetilde{R} : \mathcal{K} \times \mathcal{T} \times \mathcal{B} \times \{0,1\} \to \mathcal{B}$. The encryption produces $\widetilde{E}_K^T(P) = (C_0 \,\|\, C_1)$. We define $\widetilde{E}_K^T(P)[0] = C_0$ and $\widetilde{E}_K^T(P)[1] = C_1$ The decryption and tag-reconstruction take a bit $b$ such that it holds $\widetilde{D}_K^{T,b}(\widetilde{E}_K^T(P)[b]) = P$, for all $K, T, P, b \in \mathcal{K} \times \mathcal{T} \times \mathcal{B} \times \{0,1\}$. The tag-reconstruction produces $\widetilde{R}_K^{T,b}(\widetilde{E}_K^T(P)[b]) = \widetilde{E}_K^T(P)[b \oplus 1]$ for all tuples of inputs.
The ideal tweakable forked permutation $\widetilde{\Pi}$ encrypts messages $P$ under two independent permutations $\widetilde{\pi}_0, \widetilde{\pi}_1 \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{B})$, and outputs $(C_0 \,\|\, C_1)$ as $C_b \leftarrow \widetilde{\pi}_b(P)$, for $b \in \{0,1\}$. We define $\widetilde{\mathsf{ForkedPerm}}(\mathcal{T}, \mathcal{B})$ for the set of all tweakable forked permutations with tweak space $\mathcal{T}$ and block space $\mathcal{B}$. The security definition is then defined simply as the maximal advantage to distinguish an instance from the ideal:

**Definition 2** (PRTFP Advantage)**.** Let $\mathcal{K}$ be a non-empty set and $\mathcal{B}$, and $\mathcal{T}$ be message and tweak space, respectively. Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{B} \times \{0, 1\} \to (\mathcal{B})^2$ denote a tweakable forked permutation. Let $\widetilde{\Pi} \twoheadleftarrow \widetilde{\mathsf{ForkedPerm}}(\mathcal{T}, \mathcal{B})$ and $K \twoheadleftarrow \mathcal{K}$. Then, the $\mathsf{PRTFP}$ advantage of an adversary $\mathbf{A}$ w.r.t. $\widetilde{E}$ is defined as $\mathbf{Adv}^{\mathsf{PRTFP}}_{\widetilde{E}}(\mathbf{A}) \overset{\text{def}}{=} \Delta_{\mathbf{A}}(\widetilde{E}_K; \widetilde{\Pi})$.

**Iterate-Fork-Iterate.** A secure (tweakable) forkcipher could be instantiated easily from two secure independent (tweakable) permutations $\widetilde{\pi}_0$ and $\widetilde{\pi}_1$. Andreeva et al. presented the Iterate-Fork-Iterate (IFI) construction, which employs three independent permutations $\widetilde{\pi}, \widetilde{\pi}_0, \widetilde{\pi}_1 \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{B})$ to encrypt tweak-plaintext tuples $(T, P)$ as

$$\mathrm{IFI}[\widetilde{\pi}, \widetilde{\pi}_0, \widetilde{\pi}_1]^T(P) \overset{\text{def}}{=} \widetilde{\pi}_0^T\left(\widetilde{\pi}^T(P)\right) \parallel \widetilde{\pi}_1^T\left(\widetilde{\pi}^T(P)\right).$$

Their crucial idea was to instantiate those permutations with round-reduced AES to reduce the computational costs.

## 2.2 Brief Introduction of AES, KIASU-BC, and ForkAES

**The AES-128.** We try to follow the notation conventions from the AES and KIASU-BC as far as possible. The AES-128 is a substitution-permutation network over 128-bit inputs, which transforms the input through ten rounds consisting of SubBytes (SB), ShiftRows (SR), MixColumns (MC), and a round-key addition with a round key $K^i$. Before the first round, a whitening key $K^0$ is XORed to the state; the final round omits the MixColumns operation. We write $S^i$ for the state after Round $i$, and $S^i[j]$ for the $j$-th byte, for $0 \leq i \leq 10$ and $0 \leq j \leq 15$, where the byte ordering is given by:

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

We will use a similar convention for the round keys $K^i$ and their bytes $K^i[j]$, for $0 \leq i \leq 16$. Moreover, we denote by $S^{r,\mathsf{SB}}$, $S^{r,\mathsf{SR}}$, and $S^{r,\mathsf{MC}}$ also the states in the $r$-th round directly after SubBytes, ShiftRows, and MixColumns operation, respectively. More details can be found in [DR02, Nat01].

**KIASU-BC.** KIASU-BC [JNP14] is a tweakable block cipher that aimed to be as close as possible to the AES-128. It differs only in the fact that it XORs a 64-bit tweak $T$ to the topmost two rows of the state when ever a round key is XORed. We denote the tweak by $T$ and by $T[j]$, for $0 \leq j \leq 7$, the bytes of $T$. Note that the tweak bytes are ordered as

$$\begin{bmatrix} 0 & 2 & 4 & 6 \\ 1 & 3 & 5 & 7 \end{bmatrix}.$$

Moreover, we write KS for an iteration of the AES-128 key schedule.

**ForkAES.** ForkAES is a forkcipher that bases largely on KIASU-BC. It forks the state after five rounds and processes it twice to two ciphertexts $C_0$ and $C_1$. We denote the states of the first branch by $X^i \overset{\text{def}}{=} S^i$, for $5 \leq i \leq 10$, where $X^5 = S^5$ and $X^{10} = C_0$. Moreover, we denote the states of the second branch by $Y^i$, for $5 \leq i \leq 10$, where $Y^5 = S^5$ and $Y^{10} = C_1$. A schematic illustration is given in Figure 1, and more details can be found in [ARVV18].

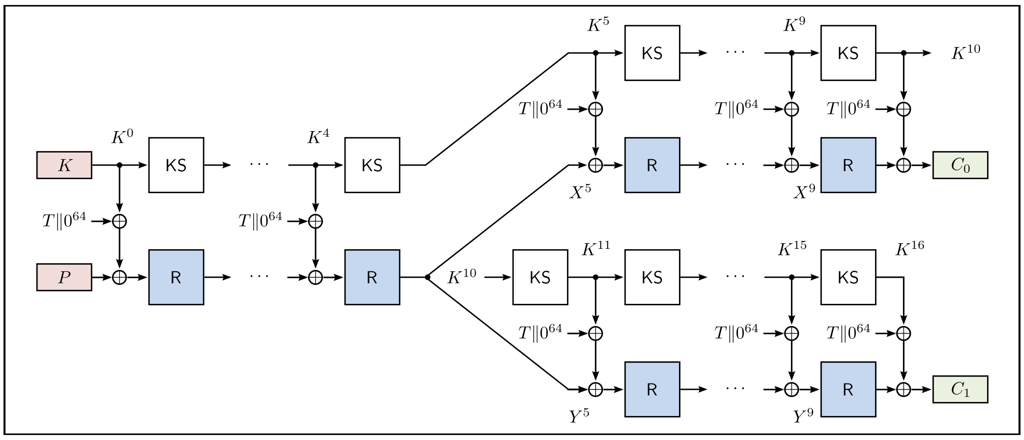

**Figure 1:** ForkAES. R denotes the round function and KS an iteration of the AES-128 key schedule.

We denote the round function by $\mathsf{R}^T_{K^i}$; i.e., $\mathsf{R}^T_{K^i}(x) =^{\text{def}} \mathsf{MC}(\mathsf{SR}(\mathsf{SB}(x))) \oplus K^i \oplus T$. We will also write $\mathsf{R}$ for the operation sequence $\mathsf{MC} \circ \mathsf{SR} \circ \mathsf{SB}$. For $i < j$, we denote by $\mathsf{R}^T_{K^{i..j}}$ the sequence of rounds $\mathsf{R}^T_{K^{i..j}} =^{\text{def}} \mathsf{R}^T_{K^j} \circ \mathsf{R}^T_{K^{j-1}} \circ \cdots \circ \mathsf{R}^T_{K^i}$. As a shortcut, $\mathsf{R}^T_{K^{i..j},L}$ denotes $\mathsf{R}^T_L \circ \mathsf{R}^T_{K^{i..j}}$.
Later, we will sometimes use the fact that linear operations can be reordered; in particular, we will reorder the MixColumns operation and the round-key addition. For this purpose, we define $\widehat{K}^r = \mathsf{MixColumns}^{-1}(K^r)$ for the corresponding round keys.

## 2.3 Properties of the AES

**Subspaces of the AES.** We adopt the notation of subspaces for the AES from Grassi et al. [GRR16]. We recall the definition of a coset from Grassi et al: given a vector space $\mathcal{W}$ and a subspace $\mathcal{V} \subseteq \mathcal{W}$. If $a$ is an element of $\mathcal{W}$, then, a coset $\mathcal{V} \oplus a$ of $\mathcal{V}$ in $\mathcal{W}$ is a subset $\mathcal{V} \oplus a = \{v \oplus a | \forall v \in \mathcal{V}\}$ We consider vectors and vector spaces over $\mathbb{F}_{2^8}^{4\times 4}$, and denote by $\{e_{0,0}, \ldots, e_{3,3}\}$ the unit vectors of $\mathbb{F}_{2^8}^{4\times 4}$, i.e., $e_{i,j}$ has a single 1 in the $i$-th row and $j$-th column. For a vector space $\mathcal{V}$ and a function $F : \mathbb{F}_{2^8}^{4\times 4} \to \mathbb{F}_{2^8}^{4\times 4}$, we let $F(\mathcal{V}) =^{\text{def}} \{F(v) | v \in \mathcal{V}\}$. For a subset $\mathcal{I} \subseteq \{1, 2, \ldots, n\}$ and a subset of vector spaces $\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_n\}$, we define $\mathcal{V}_\mathcal{I} =^{\text{def}} \bigoplus_{i\in\mathcal{I}} \mathcal{V}_i$. We adopt the definitions by Grassi et al. of four families of subspaces for the AES. For each $i \in \{0, 1, 2, 3\}$, the following spaces are defined:

- the column spaces $\mathcal{C}_i$ as $\mathcal{C}_i = \langle e_{0,i}, e_{1,i}, e_{2,i}, e_{3,i} \rangle$,
- the diagonal spaces $\mathcal{D}_i$ as $\mathcal{D}_i = \mathsf{SR}^{-1}(\mathcal{C}_i)$,
- the inverse-diagonal spaces $\mathcal{ID}_i$ as $\mathcal{ID}_i = \mathsf{SR}(\mathcal{C}_i)$, and
- the mixed spaces $\mathcal{M}_i$ as $\mathcal{M}_i = \mathsf{MC}(\mathcal{ID}_i)$.

For $\mathcal{I} \subseteq \{0, 1, 2, 3\}$, the spaces $\mathcal{C}_\mathcal{I}$, $\mathcal{D}_\mathcal{I}$, $\mathcal{ID}_\mathcal{I}$, and $\mathcal{M}_\mathcal{I}$ are defined as

$$\mathcal{C}_\mathcal{I} \overset{\text{def}}{=} \bigoplus_{i\in\mathcal{I}} \mathcal{C}_i, \qquad \mathcal{D}_\mathcal{I} \overset{\text{def}}{=} \bigoplus_{i\in\mathcal{I}} \mathcal{D}_i, \qquad \mathcal{ID}_\mathcal{I} \overset{\text{def}}{=} \bigoplus_{i\in\mathcal{I}} \mathcal{ID}_i, \quad \text{and} \quad \mathcal{M}_\mathcal{I} \overset{\text{def}}{=} \bigoplus_{i\in\mathcal{I}} \mathcal{M}_i.$$

The S-box $\mathsf{S} : \mathbb{F}_{2^8} \to \mathbb{F}_{2^8}$ of the AES has a few well-analyzed properties; here, we briefly recall two that are relevant in our later attacks.

**Property 1.** Let $\alpha \in \mathbb{F}_{2^8} \setminus \{0^8\}$ be an arbitrary but fixed input difference. Let $x, y \twoheadleftarrow \mathbb{F}_{2^8}$. Then, for $F \in \{\mathsf{S}, \mathsf{S}^{-1}\}$, it holds that $\Pr[F(x) \oplus F(x \oplus \alpha) = F(y) \oplus F(y \oplus \alpha)] = \frac{520}{2^{16}} \approx 2^{-6.98}$.

**Property 2.** Let $\alpha, \beta \in \mathbb{F}_{2^8} \setminus \{0^8\}$. Then,

$$|\{x : \mathsf{S}(x) \oplus \mathsf{S}(x \oplus \alpha) = \beta\}| = \begin{cases} 4 & \text{in 1 case} \\ 2 & \text{in 126 cases} \\ 0 & \text{in 128 cases.} \end{cases}$$

So, for any non-trivial differential $\alpha \to \beta$, there exists one input $x$ on average that satisfies the differential. This implies that for each input difference $\alpha \in \mathbb{F}_{2^8} \setminus \{0^8\}$, there exists one output difference $\beta \in \mathbb{F}_{2^8} \setminus \{0^8\}$ s. t. $\Pr[\mathsf{S}(x) \oplus \mathsf{S}(x \oplus \alpha) = \beta] = 2^{-6}$, 126 differences $\beta$ such that $\Pr[\mathsf{S}(x) \oplus \mathsf{S}(x \oplus \alpha) = \beta] = 2^{-7}$, and 128 differences $\beta$ s. t. the probability is zero. The same property also holds through the inverse S-box.

## 2.4 Boomerang and Rectangle Attacks

**Boomerangs.** Boomerang attacks [Wag99] are one form of advanced differential cryptanalysis. They allow to apply an attack from the composition of two short high-probability differentials in cases where long differentials with sufficient probability are unknown or lacking. Given a cryptographic transform $E : \mathbb{F}_2^n \to \mathbb{F}_2^n$, it is decomposed into parts $E = E_2 \circ E_1$ such that there exist a differential $\alpha \to \beta$ with probability $p$ over $E_1$ and a differential $\gamma \to \delta$ with probability $q$ over $E_2$. Often, the differentials are referred to as upper and lower differentials or trails. A boomerang distinguisher then chooses plaintext pairs $(P, P')$, with $P' = P \oplus \alpha$, and asks for the corresponding ciphertexts $(C, C')$ through $E$. It derives $D = C \oplus \delta$ and $D' = C' \oplus \delta$ to obtain the ciphertext pair $(D, D')$, and asks for the corresponding plaintext tuples $(Q, Q')$. If $Q \oplus Q' = \alpha$, then $(P, P', Q, Q')$ forms a *correct quartet*. The probability of a correct quartet is often approximated by $(pq)^2$ since the trails must hold for both pairs. The probability can be increased by considering all possible internal trails $\alpha \to \beta'$ and $\gamma' \to \delta$ as long as *both* pairs in the quartet have differences $\beta'$ and $\gamma'$ in the middle and $\beta \neq \gamma$. Hence, the simplified probability of a correct quartet increases to $(\widehat{pq})^2$ with

$$\widehat{p} = \sqrt{\sum_{\beta'} \Pr^2[\alpha \to \beta']} \quad \text{and} \quad \widehat{q} = \sqrt{\sum_{\gamma'} \Pr^2[\gamma' \to \delta]}.$$

Clearly, the attack demands that $\widehat{pq} \gg 2^{-n/2}$ for the differential to be probable. Given $N$ plaintext pairs, one expects about $N \cdot (\widehat{pq})^2$ correct quartets in an attack, but only $N \cdot 2^{-n}$ correct quartets for an ideal primitive, which yields a distinguishing event.
For the AES S-box, it follows from Property 1 for arbitrary $\alpha, \delta \in \mathbb{F}_{2^8} \setminus \{0\}$ that

$$\sqrt{\sum_{\beta'} \Pr^2\left[\alpha \xrightarrow{\mathsf{S}} \beta'\right]} \simeq 2^{-3.5} \qquad \sqrt{\sum_{\gamma'} \Pr^2\left[\gamma' \xleftarrow{\mathsf{S}} \delta\right]} \simeq 2^{-3.5}.$$

**Rectangle Attacks.** Rectangle attacks [BDK01, BDK05] are a chosen-plaintext form of the boomerang concept. The core difference of rectangles is to encrypt many plaintext $(P, P')$ with difference $\alpha$ and simply hope that some of those will form a quartet with the desired differences in the middle. Given $N$ plaintext pairs, the number of correct quartets is reduced to $N^2 \cdot 2^{-n} \cdot (\hat{p}\hat{q})^2$ by a birthday argument. The left side of Figure 2 illustrates a related-tweakey rectangle. Biham et al. presented further technical improvements to the technique in [BDK02]. The major disadvantages of rectangle compared to boomerang attacks are the higher data complexity and the large number of potential quartets that have to be handled.

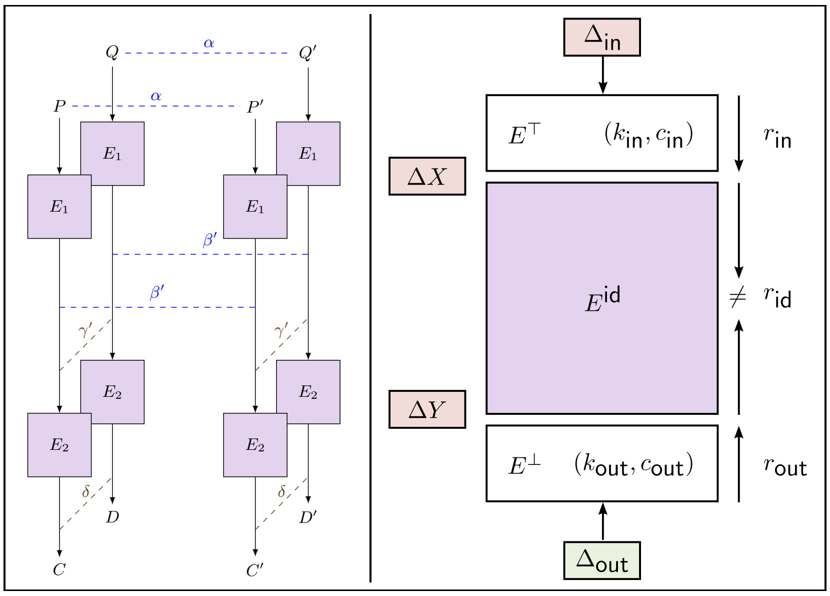

**Figure 2:** Schematic illustrations of a related-tweakey rectangle (**left**) and an impossible-differential attack (**right**).

**Verification.** Boomerangs and rectangles represent oversimplified equation systems, as was first stressed by Murphy [Mur11]. Cid et al. [CHP+18] proposed the concept of a Boomerang-connectivity Table for S-box-based ciphers as a tool that allows to identify if boomerangs can really hold. We consider their approach later in this work. For the interested reader, we note that the approach was recently refined in [BC18], and the effort for its computation was reduced in [Dun18].

## 2.5 Impossible-differential Attacks

Impossible-differential attacks have been proposed independently by Knudsen [Knu98] and Biham et al. [BBS99]. Throughout this work, we follow the generic framework by Boura et al. [BNS14]; the interested reader is referred to recent refinements by Blondeau [Blo17]. The search of an impossible-differential attack usually consists of two steps: first, one searches for a differential $\Delta X \not\rightarrow \Delta Y$ with probability zero through a sub-cipher; thereupon, one propagates the start difference $\Delta X$ backwards through $r_{\text{in}}$ rounds to an input difference $\Delta_{\text{in}}$, and the end difference $\Delta Y$ forwards through $r_{\text{out}}$ rounds to an output difference $\Delta_{\text{out}}$. The complexity can be reduced by considering $\Delta_{\text{in}}$ and $\Delta_{\text{out}}$ as vector spaces that can contain multiple allowed start and end differences.

The adversary queries plaintexts $P_i$ and constructs pairs $(P_i, P_j)$ s. t. their difference lies in $\Delta_{\text{in}}$; it asks for their corresponding ciphertexts $C_i$, and considers only ciphertext pairs $(C_i, C_j)$ whose differences fall into the space spanned by $\Delta_{\text{out}}$. Next, it guesses key bits through the $r_{\text{in}}$ inner rounds and the $r_{\text{out}}$ outer rounds. All key candidates that yield $\Delta X$ and $\Delta Y$ for any pair in the middle must be wrong and can be discarded. Boura et al. denoted by $c_{\text{in}}$ the number of bit conditions that have to be fulfilled for a pair with input difference in $\Delta_{\text{in}}$ to yield $\Delta X$. Similarly, they denoted the number of bit conditions that must be fulfilled to get from $\Delta_{\text{out}}$ to $\Delta Y$ by $c_{\text{out}}$. We denote the key sets by $\mathcal{K}_{\text{in}}$ and $\mathcal{K}_{\text{out}}$. The number of key bits involved is denoted by $k_{\text{in}}$ and $k_{\text{out}}$ respectively. The right image of Figure 2 comprises the notations of an impossible-differential attack.

The number of pairs needed to filter is chosen such that the probability of a key to survive is low. Following [Blo17, Knu98], let $T_K$ be the random variable that counts the number of pairs that allow discard key candidate $K$. Knudsen [Knu98] assumed that $T_K$ follows a binomial distribution with parameters $(N, p = 2^{-(c_{\mathsf{in}}+c_{\mathsf{out}})})$ that a pair leads to the impossible differential for a given key, the probability that this key candidate survives all pairs can be approximated by

$$p_{\mathsf{survive}} = \Pr\left[T_K = 0\right] = \binom{N}{0} \cdot p^0 \cdot (1-p)^N = \left(1 - 2^{-(c_{\mathsf{in}}+c_{\mathsf{out}})}\right)^N .$$

This probability can be approximated by $e^{-pN}$. The complexities are given by:

- Data: $C_N \cdot N$. For obtaining the necessary number of pairs $N$, Boura et al. [BNS14] formulated the following complexity, based on the limited birthday problem:

$$C_N = \max\left\{ \min_{\Delta \in \{\Delta_{\mathsf{in}}, \Delta_{\mathsf{out}}\}} \left\{ \sqrt{N \cdot 2^{n+1-|\Delta|}} \right\}, N \cdot 2^{n+1-|\Delta_{\mathsf{in}}|-|\Delta_{\mathsf{out}}|} \right\}. \tag{1}$$

- Memory: $N$ pairs;
- Time:

$$C_N \cdot C_E + \left(1 + \frac{2^{|k_{\mathsf{in}} \cup k_{\mathsf{out}}|}}{2^{c_{\mathsf{in}}+c_{\mathsf{out}}}}\right) \cdot N \cdot C_{E'} + 2^{k-\alpha} \cdot C_E. \tag{2}$$

The number of plaintext pairs is chosen s. t. $N$, the expected number of the ciphertext pairs with difference in $\Delta_{\mathsf{out}}$ fulfills that $p_{\mathsf{survive}} \leq 2^{-\alpha}$; so, the attack reduces the key spaces by $\alpha$ bits on average. In the most conservative fashion, $N$ is chosen to be smaller than $2^{-|k_{\mathsf{in}} \cup k_{\mathsf{out}}|}$, so that only the correct key is expected to survive. The term $C_N$ simply refers to the complexity of finding $N$ pairs with ciphertext difference in $\Delta_{\mathsf{out}}$. $C_E$ is the cost for evaluating the primitive, $k$ denotes the key size, and $C_{E'}$ the costs of the partial encryption to detect impossible differentials.
One can consider multiple impossible differentials to employ the same data for multiple impossible trails. Boura et al. point out that this strategy affects only the first term, $C_N$, but not the further terms of the memory complexity.
Boura et al. aimed at providing generic formulas for the complexities of impossible-differential attacks. At FSE 2016, Derbez pointed out cases where the generic time-complexity calculation was not correct [Der16]; those counter-examples considered optimized attacks wherein the key was recovered part by part. In their follow-up work [BLNS18], Boura et al. addressed Derbez' findings and emphasized that also given their generic formulas, the exact complexity of each attack needs to be carefully computed. Later in this work, we will refrain from employing those optimizations. Therefore, the complexity calculations for our attacks should be sub-optimal, but circumvent the pitfalls pointed out by Derbez.

## 3 Rectangle Cryptanalysis

This section presents two related-tweak rectangle attacks: Our primary focus is on a related-key attack on ten rounds in Section 3.1. However, this attack requires a specific type of difference $K^5 \oplus K^{11} \in \mathcal{D}_1$ between the round keys $K^5$ and $K^{11}$ that applies to a fraction of approximately $2^{32}$ keys. We describe an attack on nine rounds that works on the full key space afterwards in Section 3.2.

**Notation.** Given two plaintext-tweak tuples $(P, T)$ and $(P', T')$ that are encrypted to $(C_0, C_1)$ and $(C'_0, C'_1)$, respectively, we denote by $X = Y$ and $X' = Y'$ their states in the middle after five rounds. We define $\Delta X^r = X^r \oplus X'^r$ for the differences between the states after Round $r$, that lead to $C_0$; similarly, we define $\Delta Y^r = Y^r \oplus Y'^r$ for the

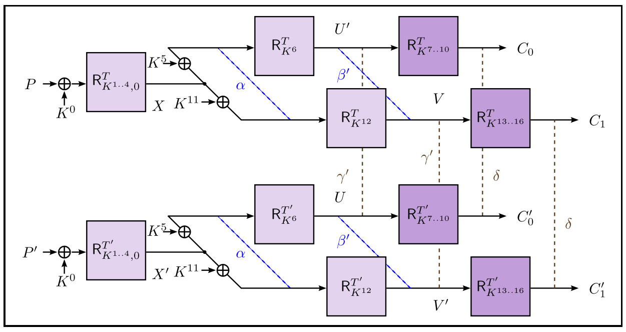

**Figure 3:** Schematic illustration of our weak-key rectangle attack on ten-round ForkAES.

differences of the states that lead to $C_1$. So, we consider two differential trails: $(\Delta X = \Delta Y) \to (\Delta C_0, \Delta C_1)$. We use $\Delta X = \Delta Y$ interchangeably for the differences in their states in the middle. This is illustrated in Figure 3.

## 3.1 Weak-key Attack on Ten Rounds

**Top Differentials.** From a high-level perspective, the top trail covers Round 6 without the final key and tweak addition, whereas the bottom trail consists of the key and tweak addition at the end of Round 6 as well as Rounds 7 through 10. The top trail concerns the differential between $X$ and $Y$, i.e., the differential path of the first portion of the trail from $X \to C_0$ and $Y \to C_1$. Clearly, this difference stems purely from the subkey differences. In contrast to the pure AES or KIASU-BC, the forking step guarantees that the difference between the inputs of Round 6 and Round 11 is the same for every plaintext, because the subkeys do not change from encryption to encryption. If the input to Round 11 would be the output of Round $10 \oplus K^{11}$, it would not be guaranteed that the difference would be the same for every encryption. The bottom trail then considers the difference between $X_i$ and $X_j$ as well as between $Y_i$ and $Y_j$. There, the differences stem from two different plaintexts, but the key difference is zero in the bottom trail.
For the top trail, we assume that the key difference of $K^5 \oplus K^{11} \in \mathcal{D}_1$, i.e., only the second diagonal is active, as illustrated on the left side of Figure 4. Since we have 32 conditions, we expect the fraction of the key space for which the attack holds to consist of approximately $2^{32}$ elements. The probability that the truncated top trail has the same $\widehat{\beta}$ in both pairs of approximately $2^{-4 \cdot 6.98} \simeq 2^{-28}$. With probability $2^{-128}$, we obtain a chosen difference in the middle after the MixColumns operation in Round 6, where we choose the start difference of the bottom trail.

**Bottom Differentials.** The bottom trail is shown on the right side of Figure 4. There are four active S-boxes at the start of Round 6. Since we need only equal differences $\gamma$, their probability is approximately $(2^{-6.98})^4 \approx 2^{-28}$ for both pairs. Moreover, we need a certain difference so that $\Delta X^{7,\mathsf{MC}}[0] = \Delta T[0]$, which holds with probability $1/(2^8 - 1) \simeq 2^{-8}$. Then, both pairs pass through Round 8 with probability one and have a difference of $\Delta T[0]$ at the end of Round 8. We guess $2 \cdot 8$ key bits of $\widehat{K}^9[0]$ and $\widehat{K}^{15}[0]$ in Round 9. We guess $2 \cdot 32$ key bits in $\widehat{K}^{10}[0, 7, 10, 13]$ and $\widehat{K}^{16}[0, 7, 10, 13]$. So, the probability for the bottom trail is approximately $2^{-28-8} = 2^{-36}$ for both pairs. The difference $\delta$ in Figure 3 corresponds to the differences $\Delta X^8$ and $\Delta Y^8$ at the end of our distinguisher.

**Initial Preparation.** We define a linear mapping $F : \mathbb{F}_{2^8}^{4 \times 4} \to \mathbb{F}_2^{96}$ of rank 96, s. t. $F(\Delta X^{10,\mathsf{SR}}) = 0^{96}$. So, we can identify later ciphertext pairs $C_i, C'_i$ with difference $\delta$ from collisions of the form $F(C_i) = F(C'_i)$ using two evaluations of $F$ per text instead

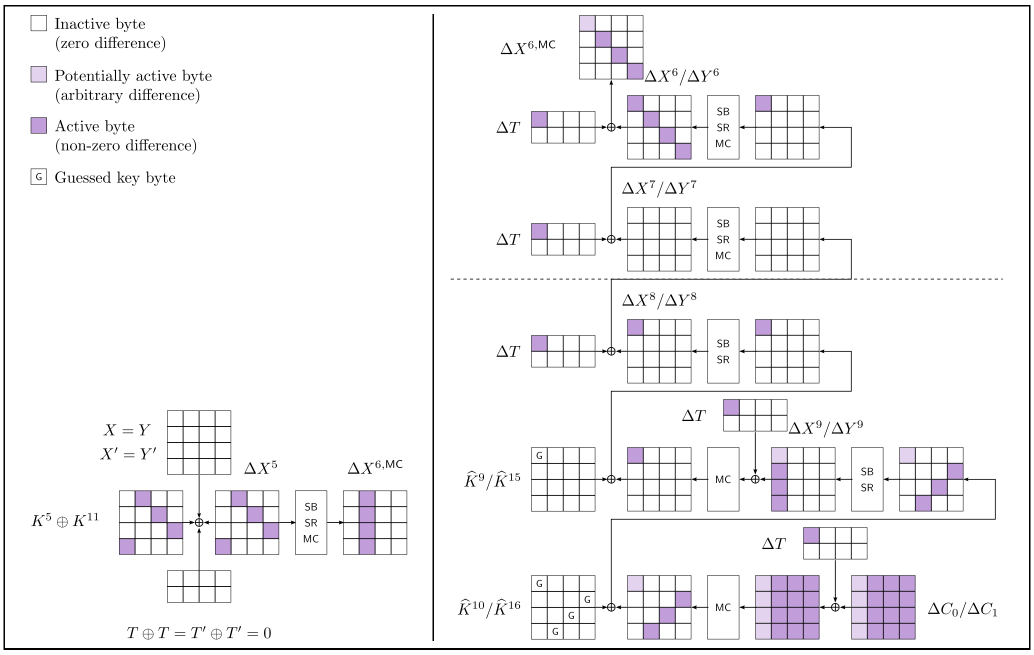
Inactive byte (zero difference)
Potentially active byte (arbitrary difference)
Active byte (non-zero difference)
G Guessed key byte
$X = Y$
$X' = Y'$
$\Delta X^5$
$\Delta X^{6,\mathsf{MC}}$
$K^5 \oplus K^{11}$
SB SR MC
$T \oplus T = T' \oplus T' = 0$
$\Delta X^{6,\mathsf{MC}}$
$\Delta X^6/\Delta Y^6$
$\Delta T$
$\Delta X^7/\Delta Y^7$
$\Delta X^8/\Delta Y^8$
SB SR
$\Delta X^9/\Delta Y^9$
$\widehat{K}^9/\widehat{K}^{15}$
$\widehat{K}^{10}/\widehat{K}^{16}$
MC
$\Delta C_0/\Delta C_1$

**Figure 4:** Trails of our ten-round rectangle attack. **Left:** Top trail over Round 6. **Right:** Bottom trail, covering the round-key and tweak addition in Round 6 and Rounds 7 through 10. The operations below the dashed horizontal line are considered during the key recovery.

of comparing all differences. In general, such a mapping $F(x) =^{\text{def}} \mathbf{F} \cdot x$ can be defined by a matrix $\mathbf{F} \in \mathbb{F}_2^{128\times128}$ s. t. $\textsc{rank}(\mathbf{F}) = 96$ and $\Delta X^{10,\mathsf{SR}} \in \textsc{kernel}(\mathbf{F})$. $F$ can be chosen efficiently and almost arbitrarily as long as the property holds. For ForkAES and our difference, this task reduces to getting a vector of the 12 inactive bytes in $\Delta X^{10,\mathsf{SR}}$, which can be realized with a few operations, e.g., with three shifts, four XORs, and four ANDs per ciphertext. A code listing is given in Appendix A.

**Further Preparation Steps.** We can perform a preparation step that will save computational effort later in the attack. Let $x = X^{10,\mathsf{SB}}[0,7,10,13]$, $x' = X'^{10,\mathsf{SB}}[0,7,10,13]$, $k^9 = \widehat{K}^9[0]$, and $k^{10} = \widehat{K}^{10}[0,7,10,13]$ be short forms. We further construct a hash map

$$\mathcal{H} : \mathbb{F}_{2^8} \times \mathbb{F}_{2^8} \times \mathbb{F}_{2^8}^4 \times \mathbb{F}_{2^8}^4 \to \left(\mathbb{F}_{2^8}^5\right)^*$$

s. t. for all inputs $(T[0], T'[0], x, x')$, $\mathcal{H}$ returns exactly those keys $(k^9, k^{10})$ that map $x$ and $x'$ to a zero difference at $\Delta X^{8,\mathsf{MC}} = 0^{4\times8}$. We obtain:

- A single fixed difference $\alpha = \Delta T[0]$;
- The difference $\alpha$ maps to 127 non-zero differences $\beta \in \mathbb{F}_{2^8}$ due to Property 2.
- The 127 differences $\beta$ are mapped uniquely to 127 differences $\gamma$ by MixColumns.
- $X^{10}[0,7,10,13]$ and $X'^{10}[0,7,10,13]$ are fixed and possess a fixed difference $\delta$.

The trail contains 32 bit conditions that have to be fulfilled. So, $\mathcal{H}$ maps to approximately $2^8$ suggestions of 40 key bits on average. The same map $\mathcal{H}$ can be used also to obtain suggestions for $\widehat{K}^{15}[0]$ and $\widehat{K}^{16}[0,7,10,13]$ from inputs $y = Y^{10,\mathsf{SB}}[0,7,10,13]$, $y' = Y'^{10,\mathsf{SB}}[0,7,10,13]$, $T[0]$, and $T'[0]$.

**Attack Steps.** The steps in the attack are as follows:

1. Initialize an empty list $\mathcal{Q}$. Initialize a zeroed list $\mathcal{K}$ of byte counters for the 40 key bits $\widehat{K}^9[0], \widehat{K}^{10}[0, 7, 10, 13]$. Initialize another zeroed list $\mathcal{L}$ of byte counters for the 40 key bits $(\widehat{K}^{15}[0], \widehat{K}^{16}[0, 7, 10, 13])$.
2. Precompute $\mathcal{H}$.
3. Choose an arbitrary base tweak $T \in \mathbb{F}_{2^8}^{2\times 4}$. Construct $2^8$ sets $\mathcal{S}^i$ from iterating over all values of $T[0] = i$. For each set, choose $2^s$ plaintexts $P$. All texts in a set use the same tweak value $T$. Ask for their $2^{s+8}$ encryptions $(T, C_0, C_1)$ from an encryption oracle. Invert the final tweak addition, and the final MixColumns operation for each output tuple $(C_0, C_1)$.
4. Process all ciphertexts by $F$ to obtain $Q_0$ and $Q_1$ from $C_0$ and $C_1$; we define $Q_b = F(\mathsf{MC}^{-1}(T \oplus C_b))$, for $b \in \{0, 1\}$. store $(T, Q_0, Q_1)$ into buckets of $\mathcal{Q}$.
5. Only consider pairs of tuples $(T, Q_0, Q_1)$ and $(T', Q'_0, Q'_1)$ if $T[0] \neq T'[0]$, $F(Q_0) = F(Q'_0)$ and $F(Q_1) = F(Q'_1)$. We call such pairs of tuples with our desired property **quartets**. Discard all tuples that do not form quartets.
6. For each quartet:

   6.1 Lookup from $\mathcal{H}$ the suggestions of the 40 key bits $\widehat{K}^9[0], \widehat{K}^{10}[0, 7, 10, 13]$ from $\Delta T[0] = T[0] \oplus T'[0]$, and the states $X^{10}[0, 7, 10, 13]$ and $X'^{10}[0, 7, 10, 13]$. We expect $2^8$ suggestions on average. For each suggestion, increment the corresponding counter in $\mathcal{K}$.

   6.2 Similarly, lookup from $\mathcal{H}$ the suggestions for the 40 key bits $\widehat{K}^{15}[0]$, $\widehat{K}^{16}[0, 7, 10, 13]$. We expect $2^8$ suggestions on average. For each suggestion, increment the corresponding counter in $\mathcal{K}$.
7. Output the keys in $\mathcal{K}$ and $\mathcal{L}$ in descending order of their counters.

**Data Complexity.** The adversary chooses $2^8$ sets of $2^s$ texts each. So, it can combine all texts of each pair of distinct sets to pairs, which yields $\binom{2^8}{2} \simeq 2^{15}$ combinations of sets of $2^{2s}$ text pairs, or $2^{2s+15}$ pairs. The top trail has a probability of $2^{-28}$, which yields $2^{2s-13}$ pairs in the middle. The probability that a pair of tuples forms one of our desired differences $\gamma'$ for both pairs at the start of the bottom trail is approximately by $2^{-128}$. So, we can expect $2^{2s-141}$ quartets. Since the bottom trail has a probability of $2^{-36}$ until $\Delta X^{8,\mathsf{MC}}$, we expect $2^{2s-177}$ correct quartets that share a four-byte difference at the bottom. Choosing $s = 91$ yields $2^5$ quartets on average, and requires $2^{s+8} = 2^{99}$ chosen plaintexts-tweak queries.

**Computational Complexity.** In Step (2), the adversary precomputes $\mathcal{H}$ with $2^{80}$ computations of twice a quarter round of the AES, which can be approximated by $2^{80} \cdot 2/15 \cdot 1/4 \simeq 2^{75.1}$ computations of ForkAES. In Step (3), the adversary has to ask for the encryptions of $2^{s+8}$ plaintext-tweak tuples from an oracle. Each encryption requires the evaluation of 15 AES rounds. Step (4) costs $2 \cdot 2^{s+8}$ evaluations of $F$, which we estimate by a ForkAES encryption equivalent. Moreover, we need $2 \cdot 2^{s+8} \cdot (s+8)$ memory accesses to $\mathcal{Q}$. This step yields $2^{2s+15} \cdot 2^{-192} = 2^{2s-177}$ wrong quartets plus $2^{2s-177}$ correct quartets on average. So, we expect $2^{2s-176}$ quartets on average.
For each surviving quartet, it requires $2 \cdot 2^8$ memory accesses to $\mathcal{H}$ plus $2 \cdot 2^8$ memory accesses to $\mathcal{K}$ and $\mathcal{L}$ on average. We expect an average sum of all counters of $2^8 \cdot 2^{2s-176} = 2^{14}$ in each of both lists, distributed normally over the keys. For $s = 91$, we expect $2^{14}$

**Table 2:** Histogram of the Boomerang-connectivity Table and our setting in the ten-round rectangle attack for the AES S-box.

| | #Solutions | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 2 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 20 | 256 | Average |
| BCT [CHP+18] | 32,640 | 31,620 | 255 | 510 | | | | | | | | 511 | 1.035 |
| Our setting | 30,572 | 15,373 | 11,531 | 4,611 | 1,957 | 640 | 252 | 60 | 19 | 8 | 2 | 511 | 2.014 |

counters over the 40 key bits in each list, but at most only a few that have at least 16 counts or more. So, we obtain an attack complexity of

$$2^{75.1} + 2^{s+8} + 2 \cdot 2^{s+8} \simeq 2^{100.6} \text{ Encryptions and}$$
$$2 \cdot 2^{s+8} \cdot (s+8) + 2^{2s-176} \cdot 2 \cdot 2^8 + 2^{2s-176} \cdot 2 \cdot 2^8 + 2^{14} \simeq 2^{106.63} \text{ Memory accesses.}$$

**Memory Complexity.** The attack needs $2^{80} \cdot 2^8 \cdot 40$ bits, or approximately $2^{90.33}$ bytes for $\mathcal{H}$. In addition, one needs $2^{80}$ byte counters for the keys in $\mathcal{K}$ and $\mathcal{L}$. Finally, the list $\mathcal{Q}$ requires $2^{s+8} \cdot (2 \cdot 16 + 8) < 2^{s+13.33} < 2^{104.4}$ bytes, which dominates the memory complexity.

**Computational Verification.** We aimed to verify the middle step of the ten-round attack. We can observe that, among the four active S-boxes in Round 6 in the top trail and the four active S-boxes in Round 7 in the bottom trail, only a single S-box, $\Delta X^6[5]$, is shared in the middle; more detailed, we consider the mapping

$$\alpha = X^5[9] \oplus Y^5[9] \xrightarrow{\mathsf{SB,SR,MC}} \Delta X^6[5] \xrightarrow{\mathsf{AT}[K^6[5],T[3]],\mathsf{SB,SR}}$$
$$\Delta X^{7,\mathsf{SR}}[1] \xrightarrow{\mathsf{MC,AT}[K^7[0],T[0]]} \Delta T[0] = \delta.$$

The inverse MixColumns matrix defines a unique mapping $\Delta X^{7,\mathsf{SR}}[1] = \texttt{0x09} \cdot \Delta T[0]$ in $\mathbb{F}_{2^8}$. The remaining active S-boxes are independent and can be assumed to have probability approximately $2^{-3.5}$ per pair and S-box. We evaluated the approach illustrated on the right side of Figure 5: over all $(2^8)^3$ combinations of $(x, \alpha, \delta)$, obtain $(w, z)$ from passing $(x, x \oplus \alpha)$ into two applications of the S-box $\mathsf{S}$; derive $w' = w \oplus \delta$ and $z' = z \oplus \delta$ and compute $(x', y')$ from two applications of $\mathsf{S}^{-1}$. Increase the counter in a list $\mathcal{L}[\alpha][\delta]$ if $x' \oplus y' = \alpha$.
The results are given in Table 2. The number of solutions is on average

$$\left|\left\{x \in \mathbb{F}_{2^8} : \mathsf{S}^{-1}(\mathsf{S}^{-1}(\mathsf{S}(\mathsf{S}(x)) \oplus \delta) \oplus \mathsf{S}^{-1}(\mathsf{S}^{-1}(\mathsf{S}(\mathsf{S}(x \oplus \alpha)) \oplus \delta) = \alpha\right|\right\} \simeq 2.01.$$

So, we expect a probability of approximately $2.01/2^8 \simeq 2^{-7}$ for each $(\alpha, \delta) = (X^5[9] \oplus Y^5[9], \Delta T[0])$. Multiplying this probability with $(2^{-3.5 \cdot 3})^2$ for the three further active S-boxes $\Delta X^5[3, 4, 13]$ in the top trail and with $(2^{-3.5 \cdot 3})^2$ for the three further active S-boxes $\Delta X^6[0, 10, 15]$ in the bottom trail yields a probability of the middle step of approximately $2^{-49}$. So, the probability in practice might be $2^7$ times higher than the theoretical expectation; we employ the conservative estimation in our complexity calculation.

## 3.2 A Nine-round Attack on The Full Key Space

From our attack above, we can derive in straight-forward manner a more efficient attack on the full key space on nine rounds, which omits Round 6 and inherits the remaining steps. For the sake of clarity, we define that the key schedule produces the round keys $K^0$,

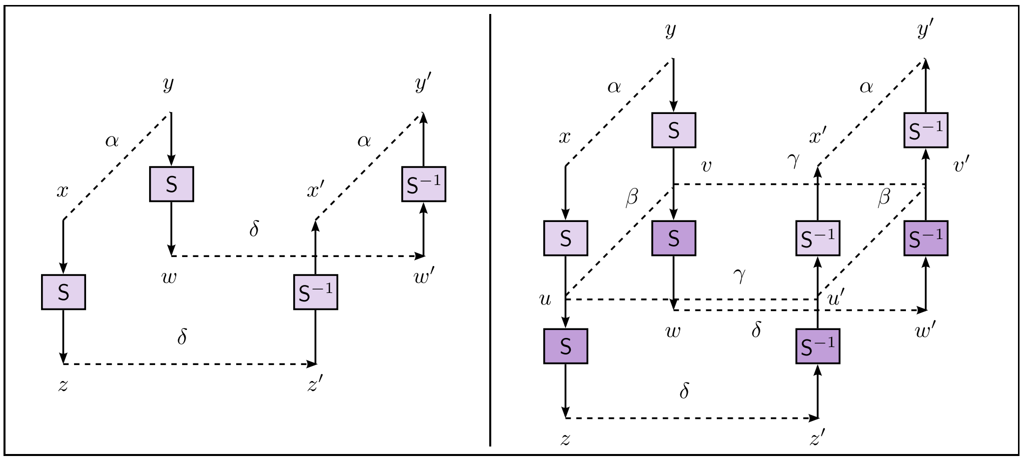

**Figure 5: Left:** The setting of Cid et al.'s Boomerang-connectivity Table; **Right:** Our setting for verifying the middle phase, where $\alpha = X^5[9] \oplus Y^5[9]$, $\gamma = \Delta X^6[5]$, and $\delta = \Delta X^{7,\mathsf{SR}}[1]$.

$\ldots$, $K^{14}$; we say that the computation from the forking state $X$ to $C_0$ employs the round keys $K^5$ through $K^9$, and the computation from $X$ to $C_1$ employs $K^{10}$ through $K^{14}$. An overview and the bottom trail are visualized in Figure 6.

**Top Differential.** Again, we construct $2^8$ sets of $2^s$ plaintext-tweak tuples as before where the sets differ again in the value of $T[0]$ and all plaintexts in a set share the same tweak. The top trail is reduced to the key addition of $K^5$ for the $X$ branch and $K^{10}$ to the $Y$ branch. So, $\alpha = \beta' = K^5 \oplus K^{10}$, which holds with probability one for each pair. The adversary simply collects pairs and waits for the difference at the beginning of the bottom trail which occurs with probability of $2^{-128}$. We obtain approximately $2^{2s+15}$ quartets, among which $2^{-128}$ are expected to have the difference $\gamma'$ in the middle, which yields $2^{2s-113}$ quartets at the forking step.

**Bottom Differentials.** The bottom trail is shown on the right side of Figure 6; it differs from that one used in the previous attack in the fact that it covers Rounds 6 through 9 here. The probability that a quartet follows the bottom trail until $\Delta X^{8,\mathsf{MC}}$ is $2^{-36}$; so, we expect $2^{2s-149}$ correct quartets. At the end, we consider the round keys $\widehat{K}^9[0,7,10,13]$, $\widehat{K}^8[0]$, $\widehat{K}^{14}[0,7,10,13]$ and $\widehat{K}^{13}[0]$. The attack steps are analogous to those above.

**Data Complexity.** We choose $2^8$ sets of $2^s$ texts each, and expect $2^{2s-149}$ correct quartets. Using $s = 77$ yields $2^5$ correct quartets on average, and needs $2^{85}$ plaintext-tweak tuples.

**Computational Complexity.** In Step (3), the adversary asks for the encryption of $2^{s+8}$ plaintext-tweak tuples from an oracle, which requires 13 AES rounds each. Step (4) costs $2 \cdot 2^{s+8}$ evaluations of $F$ and $2 \cdot 2^{s+8} \cdot (s+8)$ memory accesses. This step yields $2^{2s+15} \cdot 2^{-192} = 2^{2s-177}$ wrong quartets plus $2^{2s-149}$ correct quartets on average. So, we expect $2^5$ quartets on average. So, there is no need to precompute $\mathcal{H}$ here, but the adversary can instead test the keys on the fly, for $2 \cdot 2^5$ states of $2^{40}$ keys, of 1/4 of the state through two out of 13 rounds of the construction.
For each surviving quartet, it requires $2 \cdot 2^8$ memory accesses to $\mathcal{H}$ plus $2 \cdot 2^8$ memory accesses to $\mathcal{K}$ and $\mathcal{L}$ on average. We expect an average sum of all counters of $2^8 \cdot 2^{2s-149} = 2^{13}$ in each of both lists, distributed normally over the keys. For $s = 77$, we expect $(2^{-23} \cdot 2^8) + 2^5 \cdot 2^8 \simeq 2^{13}$ counters over the 40 key bits on average. We can expect that

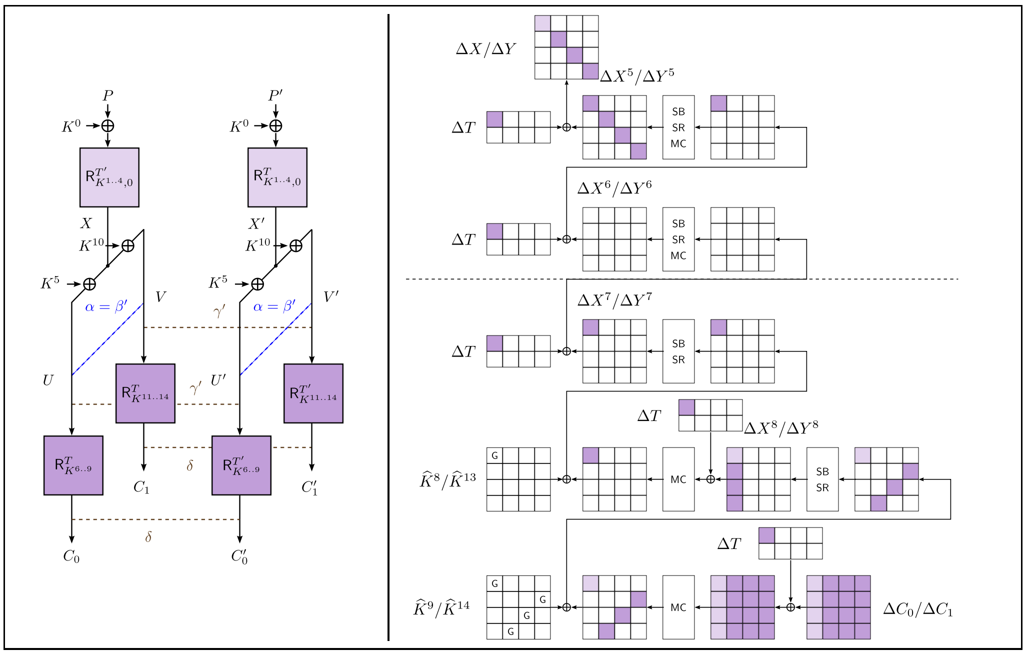

**Figure 6:** Overview (**left**) and bottom trail (**right**) of our nine-round rectangle attack. The operations below the dashed horizontal line are considered during the key recovery.

the correct keys have a significantly higher number of counts. So, we obtain

$$2 \cdot 2^5 \cdot 2^{40} \cdot \frac{1}{4} \cdot \frac{2}{13} + 2^{s+8} + 2 \cdot 2^{s+8} \simeq 2^{86.6} \text{ Encryptions and}$$
$$2 \cdot 2^{s+8} \cdot (s+8) + 2 \cdot 2^{2s-177} \cdot 2 \cdot 2^8 + 2 \cdot 2^5 \cdot 2^8 \simeq 2^{92.4} \text{ Memory accesses.}$$

**Memory Complexity.** The attack needs again $2^{80}$ byte counters for the keys in $\mathcal{K}$ and $\mathcal{L}$; $\mathcal{Q}$ needs $2^{s+8} \cdot (2 \cdot 16 + 8) < 2^{s+13.33} \simeq 2^{90.4}$ bytes of memory, which dominates the memory complexity.

## 4 Impossible-differential Cryptanalysis

**Model.** This section outlines a related-tweak impossible-differential attack on nine-round ForkAES. Note that the tweak is public; hence, our analysis is again in the single-secret-key model. The adversary queries plaintext-tweak tuples $(P_i, T_i)$ and obtains the corresponding ciphertexts $(C_{i,0}, C_{i,1})$ from an oracle that holds a random secret key. The goal of the attack described hereafter is to identify the secret key by filtering out all wrong keys more efficiently than by exhaustive search.

**Notation.** The notation is similar to that in the previous section. We assume that $(C_0, C_1)$ are returned after nine rounds of encryption. So, compared to the full ForkAES, we omit only the final AES round and key-schedule iteration in both branches, as in the rectangle attack in Section 3.2. The states $X = Y$ are still equal after almost five rounds, without the key addition after Round 5. In the following, we will consider two differential trails: $(\Delta X, \Delta Y) \to (\Delta C_0, \Delta C_1)$, where $\Delta X = \Delta Y$.

**Impossible Differentials.** The high-level idea is straight-forward: The adversary queries plaintexts under distinct tweaks and waits for tuples $(C_{i,0}, T_i)$ and $(C_{j,0}, T_j)$. It changes a

single byte in the tweaks, s. t. $T_i[0] \neq T_j[0]$. It inverts the final $\mathsf{MixColumns}$ operation and tweak addition, and uses the ciphertexts only if their difference $\Delta\widehat{C}_0$ (before $\mathsf{MixColumns}$) activates only the inverse diagonal $\mathcal{ID}_0$. It deduces five key bytes $\widehat{K}^9[0, 7, 10, 13]$ and one in $\widehat{K}^8[0]$ such that their differences cancel after the tweak addition at the end of Round 7. Then, there is a zero difference through the inverse Round 7, which leads to a single active byte in $\Delta X^{5,\mathsf{MC}}$, and to a single active diagonal at the beginning of Round 6: $\Delta X^5 \in \mathcal{D}_0$. This is illustrated on the left side of Figure 7.
Given this trail, the adversary concerns the second trail from $\Delta C_1$ backwards to $\Delta Y$. It must hold that $\Delta X = \Delta Y$ and therefore $\Delta Y^5 \in \mathcal{D}_0$. This implies that at least one of the following three cases for $\Delta Y^7$ must hold:

(1) $\Delta Y^7$ has at least one fully active column: $\Delta Y^7 \in \mathcal{C}_i$.

(2) Bytes $\Delta Y^7[1, 2, 3]$ are active.

(3) $\Delta C_1 \in \mathcal{M}_0$, i.e., is in the mixed space, generated by $\Delta Y^{9,\mathsf{SR}} \in \mathcal{ID}_0$. Note that this event can be tested in $O(1)$.

In Case (3), it holds that $\Delta Y^{9,\mathsf{SR}} \in \mathcal{ID}_0$, which means that the $\Delta Y$ trail is similar to the $\Delta X$ trail. So, we have a differential that can strongly reduce the key space, e.g., using the rectangle distinguisher on nine rounds from the previous section. However, this section tries to exploit a different distinguisher with lower data complexity and does not have to wait for such an event.
In the former two cases, the columns 1, 2, and 3 of $\Delta Y^7$ are either completely active or completely inactive. So, the adversary can guess eight bytes of $\widehat{K}^{14}$ that are mapped to one of those columns and can filter out all key guesses where one of those columns would become partially active. Hereafter, we prove this property briefly.

**Theorem 1.** Let $\Delta X^7 \in \mathcal{C}_0 \cap \mathcal{D}_0$ and let $\Delta X^7[0] = \Delta T[0]$. Let $\mathcal{I} \subsetneq \{0, 1, 2, 3\}$. Then, one of the following statements holds: (1) $\Delta Y^{7,\mathsf{MC}} = 0$; (2) $\Delta Y^{7,\mathsf{MC}} \in \mathcal{M}_{\mathcal{I}}$.

*Proof.* Let us consider the top trail $\Delta X$: ff $\Delta X^7 \in \mathcal{C}_0 \cap \mathcal{D}_0$ holds, then only $\Delta X^7[0]$ is active. Since $\Delta X^7[0] = \Delta T[0]$, it holds that $\Delta X^{7,\mathsf{MC}} = 0$. Tracing it backwards implies that $\Delta X^{6,\mathsf{MC}}$ is also active only in the 0-th byte, which has difference $\Delta T[0]$. This yields $\Delta X^5 \in \mathcal{D}_0$, and leads to $\Delta X \in \mathcal{D}_0$, as shown on the left side of Figure 7. It may hold that $\Delta X[0] = 0$, though.
Since both $Y$ and $Y'$ use the same key difference, and $X = Y$ and $X' = Y'$ hold, we have that $\Delta Y = \Delta X$; since the tweak addition is linear, it follows that $\Delta Y^5 = \Delta X^5$, i.e., $\Delta Y^5 \in \mathcal{D}_0$, from which $\Delta Y^{6,\mathsf{MC}} \in \mathcal{C}_0$ follows. So, at least one byte of $\Delta Y^{6,\mathsf{MC}}$ must be active, but all bytes can be active. The round tweakey addition to $\Delta Y^6$ then influences only the difference in $\Delta Y^6[0]$. After Round 7, the leftmost column may have either a single active byte in $\Delta Y^7[0]$, three active bytes $\Delta Y^7[1, 2, 3]$, or four active bytes, due to the tweak addition at the end of Round 7. In contrast, Columns 1, 2, and 3 of $\Delta Y^7$ have each either a zero difference, or are fully active. So, if any of those columns is only partially active in the backward computations from $\Delta C_1$ to $\Delta Y^7$ after guessing a key candidate, that candidate must be wrong. This is illustrated on the right side of Figure 7. Our claim in Theorem 1 follows. □

**Initial Steps.** As in the previous section, we can define a linear mapping $F : \mathbb{F}_{2^8}^{4\times 4} \to \mathbb{F}_2^{96}$ of rank 96 s. t. $F(\mathsf{MC}^{-1}(\Delta C_0)) = 0$. So, we can later identify ciphertext pairs $C_0, C_0'$ with difference $\delta$ from collisions of the form $F(C_0) = F(C_0')$ using two evaluations of $F$ per text instead of comparing all differences. Such a mapping can be found and evaluated

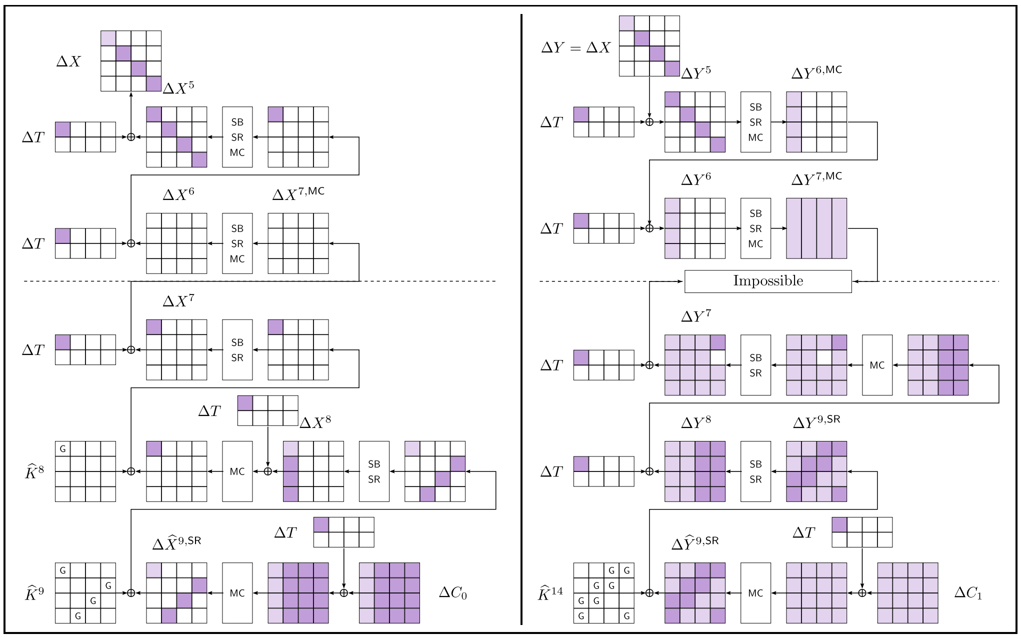

**Figure 7: Left:** $\Delta X$-trail $\Delta X \leftarrow \Delta C_0$. In the states, white bytes are inactive (zero difference); light-blue bytes are possibly active, and dark-blue bytes are active. Key bytes with G are guessed. **Right:** One variant of an impossible $\Delta Y$-trail $\Delta Y \not\leftarrow \Delta C_1$. If any of the three rightmost columns in $\Delta Y$ is active, then it is fully active. The parts below the dashed line represent the computations in the on-line phase of the attack.

efficiently and we will overestimate the effort for its evaluation by a complete ForkAES encryption equivalent. For simplicity, we define

$$\widehat{X}^{r,\mathsf{SR}} \stackrel{\text{def}}{=} \mathsf{SR}(\mathsf{SB}(X^{r-1})) \oplus \widehat{K}^r \quad \text{and} \quad \widehat{Y}^{r,\mathsf{SR}} \stackrel{\text{def}}{=} \mathsf{SR}(\mathsf{SB}(Y^{r-1})) \oplus \widehat{K}^r,$$

and $\widehat{X'}^{r,\mathsf{SR}}$ and $\widehat{Y'}^{r,\mathsf{SR}}$ analogously. Moreover, we denote by $\widetilde{K}^r = \mathsf{SR}^{-1}(\widehat{K}^r)$.
Similarly as in our rectangle attack on ten rounds, we construct a hash map

$$\mathcal{H}_0 : \mathbb{F}_{2^8} \times \mathbb{F}_{2^8} \times \mathbb{F}_{2^8}^4 \times \mathbb{F}_{2^8}^4 \to (\mathbb{F}_{2^8}^5)^*.$$

Here, it maps $x = (T[0], T'[0], \widehat{X}^{9,\mathsf{SR}}[0,7,10,13], \widehat{X'}^{9,\mathsf{SR}}[0,7,10,13])$ to all five-byte keys that yield $\Delta X^{7,\mathsf{MC}} = 0$. Moreover, we construct a second hash map

$$\mathcal{H}_1 : \mathbb{F}_{2^8}^8 \times \mathbb{F}_{2^8}^8 \to (\mathbb{F}_{2^8}^8)^*,$$

s. t. for all inputs $x = (\widehat{Y}^{9,\mathsf{SR}}[2,3,5,6,8,9,12,15], \widehat{Y'}^{9,\mathsf{SR}}[2,3,5,6,8,9,12,15])$, $\mathcal{H}_1(x)$ returns exactly those keys $\widehat{K}^{14}[2,3,5,6,8,9,12,15] = \widetilde{K}^{14}[8,9,10,11,12,13,14,15]$ s. t. they yield one of the impossible differentials in $\Delta Y^{8,\mathsf{SR}}$.
The final tweak addition, MixColumns, and ShiftRows operation can be inverted before the lookup in $\mathcal{H}_1$ is performed; the tweak addition at the end of Round 8 does not affect the difference in $\Delta Y^{8,\mathsf{SR}}$. Hence, $\mathcal{H}_1$ does not require the tweak as input.
Note that the columns can be computed independently, and $\mathcal{H}_1$ can be built from several smaller lookup tables internally. There exist four combinations of bytes: $\Delta Y^{8,\mathsf{SR}}[i,j]$ with $(i,j) \in \{(8,15),(9,12),(10,13),(11,14)\}$. Moreover, there exist two options whether Byte $i$ or Byte $j$ has a zero difference whereas the other one must have a non-zero difference. There are eight bit conditions that have to be fulfilled for an input difference to $\mathsf{MC}^{-1}$

to be mapped to an output difference with a zero-difference byte, i.e., $2^{24}$ inputs yield a difference with a zero-byte at a given byte index. On the other hand, $2^{32} - 2^{24}$ inputs yield a non-zero difference at a given byte index. Thus, given an input $Y^{9,\mathsf{SB}}$, $\mathcal{H}_1$ returns $4 \cdot 2$ combinations of $2^{24} \cdot (2^{32} - 2^4) \simeq 2^{56}$ keys that yield the impossible differential. This can be evaluated with $4 \cdot 2$ calls to two 32-bit tables each, or 16 tables that map 32 state bits to $2^{32}$ or $2^{24}$ keys. So, $\mathcal{H}_1$ needs $8 \cdot 2^{32} \cdot 2^{32} \cdot 4$ bytes $+ 8 \cdot 2^{32} \cdot 2^{24} \cdot 4$ bytes $\simeq 2^{72}$ bytes of memory. The tables can be computed with at most $16 \cdot 2^{32} \cdot 2^{32}$ quarter rounds of the AES, or $16/13 \cdot 2^{64} \simeq 2^{64.3}$ equivalents of nine-round ForkAES.

**Steps.** The steps in the attack are as follows:

1. Initialize an empty list $\mathcal{Q}$. Further initialize a list $\mathcal{K}$ that will hold all 13-byte keys $\widehat{K}^8[0]$, $\widehat{K}^9[0, 7, 10, 13]$, and $\widehat{K}^{14}[1, 2, 5, 6, 8, 9, 12, 15]$.

2. Choose an arbitrary base tweak $T \in \mathbb{F}_{2^8}^{2\times 4}$. Construct $2^8$ sets $\mathcal{S}^i$ from iterating over all values of $T[0] = i$. For each set, choose $2^s$ plaintexts $P$. All texts in a set use the same tweak $T^i$ with $T^i[0] = i$. Ask for their $2^{s+8}$ encryptions $(T, C_0, C_1)$ from an encryption oracle.

3. For each ciphertexts, invert the final tweak addition, the final MixColumns operation, and process all ciphertexts by $F$: $Q_b = F(\mathsf{MC}^{-1}(C_b \oplus T))$, for $b \in \{0, 1\}$. Store $(T, C_0, C_1, Q_0, Q_1)$ into buckets of $\mathcal{Q}$.

4. Only consider pairs of tuples $(T, C_0, C_1, Q_0, Q_1)$ and $(T', C'_0, C'_1, Q'_0, Q'_1)$ if it holds that $T \neq T'$ and $F(Q_0) = F(Q'_0)$. Discard all other tuples. We call pairs of tuples with our desired property **quartets**.

5. For each quartet:

   5.1 Derive from $\mathcal{H}_0$ the key candidates $\widehat{K}^8[0]$ and $\widehat{K}^9[0, 7, 10, 13]$ that yield zero difference in $\Delta X^{6,\mathsf{MC}}$.

   5.2 Derive from $\mathcal{H}_1$ all key candidates for $\widehat{K}^{14}[1, 2, 5, 6, 8, 9, 12, 15]$ that yield one of our eight impossible differentials. Remove all those candidates from $\mathcal{K}$.

6. Output the 13-byte key candidates remaining in $\mathcal{K}$.

**Conditions and Data Complexity.** The adversary queries $2^8$ sets of $2^s$ texts each, which yields $2^{s+8}$ chosen plaintexts. The sets differ in their value of $T[0]$, such that all texts in the set share the same $T$. One can form $\binom{2^8}{2}$ pairs of sets each with $2^s \cdot 2^s$ pairs, which yields $\simeq 2^{2s+15}$ text pairs. The adversary guesses 13 key bytes in total: $\widehat{K}^9[0, 7, 10, 13]$, $\widehat{K}^8[0]$, and $\widehat{K}^{14}[2, 3, 5, 6, 8, 9, 12, 15]$, i.e., $13 \cdot 8 = 104$ key bits.
The adversary requires texts for which it holds that $\Delta C_0 \in \mathcal{M}_0$, which occurs with probability of approximately $p \simeq 2^{-96}$. For simplicity, we assume that $(\Delta C_0, \Delta C_1) \in \mathcal{M}_0 \times \mathcal{M}_0$ never occurs by accident. From an ideal permutation, this event would have a probability of approximately $2^{-96 \cdot 2}$ per pair. Note that this event can still occur since this property is exploited in our rectangle distinguisher from Section 3.2. However, we concern a different distinguisher here..
The probability that a key $\widehat{K}^9[0, 7, 10, 13]$ reduces the four active bytes in $\Delta X^{9,\mathsf{SR}}$ to a single active byte in $\Delta X^{8,\mathsf{MC}}$ is $2^{-24}$. Moreover, the probability that this byte is mapped to $\Delta T[0]$ in $\Delta X^7$ is $2^{-8}$. So, the probability that a key in the $\Delta X$ trail yields our desired differential is approximately $2^{-32}$. There are four options of which columns in $\Delta Y^7$ becomes partially active. There are two options in which order the two known bytes in this column are active/inactive. The probability to have one inactive byte is

$2^{-8} \cdot (1 - 2^{-8})$. Hence, the probability that a key yields an impossible differential in $\Delta Y^{7,\mathsf{MC}}$ is approximately

$$2^{-8} \cdot (1 - 2^{-8}) \cdot 4 \cdot 2 \simeq 2^{-5}.$$

So, we have a probability of approximately $2^{-37}$ that a guessed key candidate yields our desired impossible differential. Following the framework by Boura et al. [BNS14], this amounts to $c \simeq 37$ bit conditions that have to be fulfilled for a key to be filtered out, given a correct pair. From our 104 key bits, we can filter out about

$$4 \cdot 2 \cdot 2^{32} \cdot (2^{32} - 2^{24}) \cdot 2^{8} \simeq 2^{67}$$

keys with a correct pair on average. The probability for a wrong key pair to survive is

$$p_{\mathsf{survive}} = \left(1 - 2^{-c}\right)^{N},$$

where $c = c_{\mathsf{in}} + c_{\mathsf{out}} \simeq 37$ is the number of bit conditions and $N$ the number of correct pairs. For $p_{\mathsf{survive}} = 2^{-104}$, we obtain $N \approx 2^{43.2}$ correct pairs to reduce the number of keys to the correct one, plus at most a few further false-positive key candidates. So, we need about $2^{43.2}$ correct pairs to reduce the key space, which implies that $2^{43.2} \cdot 2^{96} = 2^{139.2}$ pairs are necessary to find the $2^{43.2}$ ones with our desired difference $\Delta \widehat{X}^{9,\mathsf{SR}}$. We can construct about $2^{2s+15}$ pairs, which leads to $s = 62.1$ or $C_N = 2^{s+8} = 2^{70.1}$ chosen plaintext-tweak queries. For comparison, the generic Equation (1) would yield

$$C_N = \sqrt{N \cdot 2^{n+1-|\Delta_{\mathsf{out}}|}} \simeq 2^{69.6} \text{ chosen plaintexts,}$$

where $N \simeq 2^{43.2}$ is the number of necessary pairs and $|\Delta_{\mathsf{out}}| = (2^8 - 1)^4 \simeq 2^{31.98}$. Note that the other terms in the generic Equation (1) are inapplicable here. The term $N \cdot 2^{n+1-|\Delta_{\mathsf{in}}|-|\Delta_{\mathsf{out}}|}$ described the complexity if multiple structures are required, which we do not need here. Similarly, the term $\sqrt{N \cdot 2^{n+1-|\Delta_{\mathsf{in}}|}}$ would hold only in a chosen-ciphertext attack, which is inapplicable here. Our data complexity of $C_N \simeq 2^{70.1}$ is slightly higher than the result of the generic equation since we have to consider structures of sets with equal tweaks whereas the generic equation considers an untweaked cipher.

**Computational Complexity.** The computational complexity consists of the following:

- Precompute $\mathcal{H}_0$ with $2^{80}$ computations of twice a quarter round of the AES, which can be approximated by $2^{80} \cdot 2/13 \cdot 1/4 \simeq 2^{75.3}$ encryption equivalents.
- Precompute $\mathcal{H}_1$ with $2^{64.3}$ encryption equivalents.
- Encrypt $2^{s+8}$ plaintext-tweak tuples through nine-round ForkAES.
- Invert $2^{s+8} \cdot 2$ times the final tweak addition, MixColumns, and ShiftRows operation, which can be overestimated by $2^{70.1} \cdot 2 \cdot 1/13 \approx 2^{67.5}$ encryptions.
- Apply a rank-96 linear function $F$ to all states $C_0$, which is approximated by $2^{s+8}$ ForkAES computations, or $2^{70.1} \cdot 2 \simeq 2^{70.1}$ encryptions. Moreover, we need $2 \cdot 2^{s+8} \cdot (s+8) = 2 \cdot 70.1 \cdot 2^{70.1} \simeq 2^{77.3}$ memory accesses on average with an efficient data structure. We obtain approximately $2^{2s+15-96} \simeq 2^{2s-81} = 2^{43.2}$ remaining quartets.
- For each of the $2^{43.2}$ quartets, obtain from $\mathcal{H}_0$ and $\mathcal{H}_1$ with two memory accesses each $2^{67}$ keys on average. So, we have to remove them from the list with $2^{2s-14} = 2^{110.2}$ memory accesses.
- Our attack aims at recovering 104 bit conditions of $\widehat{K}^9$ and $\widehat{K}^{14}$. So, the final term for recovering the remaining bits can be safely overestimated by recovering the remaining 64 key bits of $\widehat{K}^{14}$ with $2^{64}$ nine-round encryptions.

Applying Equation (2), the attack requires approximately

$$2^{75.3} + 2^{64.3} + 2^{70.1} + 2^{67.5} + 2^{70.1} + 2^{64} \simeq 2^{75.4} \text{ Encryptions and}$$
$$2 \cdot 2^{70.1} + 2^{77.3} + 2^{43.2} \cdot 2 + 2^{43.2} \cdot 2^{67} \simeq 2^{110.2} \text{ Memory accesses.}$$

**Memory Complexity.** The attack needs $2^{80} \cdot 2^8 \cdot 40$ bits, or approximately $2^{90.33}$ bytes for $\mathcal{H}_0$. Moreover, it requires at most $2^{72}$ bytes for the components of $\mathcal{H}_1$, and requires $2^{104}$ byte counters for the keys in $\mathcal{K}$. Finally, the attack needs $2^{s+8} = 2^{70.2} \cdot (2 \cdot 16 + 8) < 2^{s+14} = 2^{76.2}$ bytes for $\mathcal{Q}$. Hence, it needs memory for

$$2^{80} + 2^{72} + 2^{104} + 2^{76.2} \simeq 2^{104} \text{ bytes.}$$

Those parts can be optimized further; though, the purpose of this work is to demonstrate attack vectors.

## 5 Conclusion

**Conclusion.** We proposed rectangle and impossible-differential attacks on nine-round variants of ForkAES, and a rectangle attack for a fraction of approximately $2^{32}$ keys for the ten-round variant. We emphasize that our attacks do not break full ForkAES in the single-key setting; however, they reduce the security margin to at most one round. We leave room for optimizations of our attacks and can envision further angles: for instance, meet-in-the-middle can also be applicable, as has been shown for KIASU-BC [TAY16]. We are also searching for longer impossible-differential trails.
The extension of ForkAES to more rounds seems recommendable, especially in the bottom phase. Moreover, it appears interesting to investigate the security when the state is forked before the fifth round, similarly to the analysis of AES-PRF [DIS+18]. The attacks presented here would be inapplicable if the forking step would be, e.g., located two rounds earlier. In Appendix B, we briefly outline the obvious implications for the security of shifted variants. However, it remains an open problem to study in detail attacks that arise from such design changes.
More generally, the impression remains that the generic forkcipher concept is at least not as easy to instantiate as to simply fork the middle state and inherit the existing analysis of a primitive, but demands deeper analysis: Our attacks exploited that the forking step ensured that the values $X$ and $Y$ are equal in the middle: if $Y$ would be chosen differently, the difference in the middle would not be guaranteed and our attacks would not apply.

**Future Work.** Our primary future work is to verify computationally the critical parts of our proposed attacks, which includes the middle phase of our rectangle attack, the key-recovery phase of the impossible differential, and the size of the weak key space of our ten-round attack. Secondary, it is interesting to study the effect on the authenticated encryption schemes PAEF, SAEF, and fGCM. Finally, investigating the security of the generic forkcipher construction is of independent relevance.
From a constructive angle, it may also be worth to consider forkciphers with more than two forked branches, which could yield more efficient constructions. Furthermore, we can observe that the forkcipher construction could be easily transformed into a highly secure PRF when both outputs are XORed to a single output block. One can easily foresee highly secure modes of operation. We are actively working on those aspects and plan to address them in an updated version of this work.

**Acknowledgments.** We gratefully thank the forkcipher designers for their very helpful comments that helped to greatly improve this work.

## References

[ARVV18] Elena Andreeva, Reza Reyhanitabar, Kerem Varici, and Damian Vizár. Forking a Blockcipher for Authenticated Encryption of Very Short Messages. Cryptology ePrint Archive, Report 2018/916, 2018. https://eprint.iacr.org/2018/916, Version: 20180926:123554.

[BBS99] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In Jacques Stern, editor, *EUROCRYPT*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–23. Springer, 1999.

[BC18] Christina Boura and Anne Canteaut. On the Boomerang Uniformity of Cryptographic Sboxes. *IACR Transactions on Symmetric Cryptology*, 2018(3):290–310, Sep. 2018.

[BDK01] Eli Biham, Orr Dunkelman, and Nathan Keller. The Rectangle Attack - Rectangling the Serpent. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *LNCS*, pages 340–357. Springer, 2001.

[BDK02] Eli Biham, Orr Dunkelman, and Nathan Keller. New Results on Boomerang and Rectangle Attacks. In Joan Daemen and Vincent Rijmen, editors, *FSE*, volume 2365 of *LNCS*, pages 1–16. Springer, 2002.

[BDK05] Eli Biham, Orr Dunkelman, and Nathan Keller. Related-Key Boomerang and Rectangle Attacks. In *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 507–525. Springer, 2005.

[BLNS18] Christina Boura, Virginie Lallemand, María Naya-Plasencia, and Valentin Suder. Making the Impossible Possible. *J. Cryptology*, 31(1):101–133, 2018.

[Blo17] Céline Blondeau. Accurate Estimate of the Advantage of Impossible Differential Attacks. *IACR Trans. Symmetric Cryptol.*, 2017(3):169–191, 2017.

[BNS14] Christina Boura, María Naya-Plasencia, and Valentin Suder. Scrutinizing and Improving Impossible Differential Attacks: Applications to CLEFIA, Camellia, LBlock and Simon. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT (1)*, volume 8873 of *LNCS*, pages 179–199. Springer, 2014.

[CHP$^+$18] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang Connectivity Table: A New Cryptanalysis Tool. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT II*, volume 10821 of *LNCS*, pages 683–714. Springer, 2018.

[Der16] Patrick Derbez. Note on Impossible Differential Attacks. In Thomas Peyrin, editor, *FSE*, volume 9783 of *Lecture Notes in Computer Science*, pages 416–427. Springer, 2016.

[DIS$^+$18] Patrick Derbez, Tetsu Iwata, Ling Sun, Siwei Sun, Yosuke Todo, Haoyang Wang, and Meiqin Wang. Cryptanalysis of AES-PRF and Its Dual. *IACR Trans. Symmetric Cryptol.*, 2018(2):161–191, 2018.

[DL17] Christoph Dobraunig and Eik List. Impossible-Differential and Boomerang Cryptanalysis of Round-Reduced KIASU-BC. In Helena Handschuh, editor, *CT-RSA*, volume 10159 of *LNCS*, pages 207–222. Springer, 2017.

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

[Dun18] Orr Dunkelman. Efficient Construction of the Boomerang Connection Table. *IACR Cryptology ePrint Archive*, 2018:631, 2018.

[GM16] Shay Gueron and Nicky Mouha. Simpira v2: A Family of Efficient Permutations Using the AES Round Function. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT I*, volume 10031 of *Lecture Notes in Computer Science*, pages 95–125, 2016.

[GRR16] Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. Subspace Trail Cryptanalysis and its Applications to AES. *IACR Trans. Symmetric Cryptol.*, 2016(2):192–225, 2016.

[JNP14] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT (2)*, volume 8874 of *LNCS*, pages 274–288, 2014.

[KLMR16] Stefan Kölbl, Martin M. Lauridsen, Florian Mendel, and Christian Rechberger. Haraka v2 - Efficient Short-Input Hashing for Post-Quantum Applications. *IACR Trans. Symmetric Cryptol.*, 2016(2):1–29, 2016.

[Knu98] Lars Knudsen. DEAL – A 128-bit block cipher. *Complexity*, 258(2):216, 1998.

[Mur11] Sean Murphy. The Return of the Cryptographic Boomerang. *IEEE Trans. Information Theory*, 57(4):2517–2521, 2011.

[Nat01] National Institute of Standards and Technology. FIPS 197. *National Institute of Standards and Technology, November*, pages 1–51, 2001.

[Nat18] National Institue of Standards and Technology. Lightweight Cryptography. Technical report, Aug 27 2018.

[TAY16] Mohamed Tolba, Ahmed Abdelkhalek, and Amr M. Youssef. A Meet in the Middle Attack on Reduced Round Kiasu-BC. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, E99-A(10):21–34, Oct 2016.

[Wag99] David Wagner. The Boomerang Attack. In Lars R. Knudsen, editor, *FSE*, volume 1636 of *LNCS*, pages 156–170. Springer, 1999.

## A Linear Mapping $F$ Used in The Attacks

**Listing 1:** Linear mapping of $\Delta C_0$ using Intel AES-NI intrinsics.

```c
__m128i map_ciphertext(const __m128 c, const __m128 t) {
    __m128i v = _mm_xor_si128(c, t);
    v = _mm_aesimc_si128(v);
    __m128i x0 = _mm_and_si128(v, _mm_set_epi32(0, 0, 0, 0x0000FFFF));
    v = _mm_bsrli_si128(v, 1);
    __m128i x1 = _mm_and_si128(v, _mm_set_epi32(0, 0, 0, 0xFFFF0000));
    v = _mm_bsrli_si128(v, 1);
    __m128i x2 = _mm_and_si128(v, _mm_set_epi32(0, 0, 0x0000FFFF, 0));
    v = _mm_bsrli_si128(v, 1);
    __m128i x3 = _mm_and_si128(v, _mm_set_epi32(0, 0xFFFFFFFF, 0xFFFF0000, 0));
    x0 = _mm_xor_si128(x0, x1);
    x2 = _mm_xor_si128(x2, x3);
    return _mm_xor_si128(x0, x2);
}
```

## B Shifted Variants

This section considers the security of variants of ForkAES where the forking step is shifted.

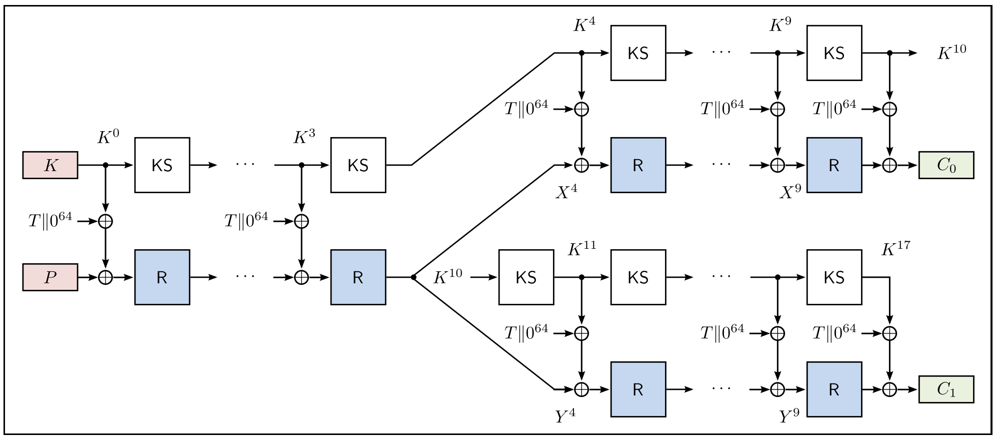

**Figure 8:** Shifted variant ForkAES[4].

**Definition.** We denote by ForkAES$[r]$ the variant where the forking step is located after the MixColumns operation of Round $r$ has been performed, i.e., directly before the AddRoundTweaKey operation with $T$ and $K^r$. So, the original ForkAES construction corresponds to ForkAES[5]. As an example, Figure 8 illustrates ForkAES[4], where the forking step is located after the MixColumns operation of Round 4. For a plausible range of $r' \in \{0, \ldots, 9\}$, ForkAES$[r']$ needs $20 - r'$ AES rounds.

**Security.** It is easy to see that the security guarantees of variants ForkAES$[r]$ with $r > 5$ can be at most those of ForkAES. In general, we can observe two informal statements:

**Statement (1):** The security guarantees of ForkAES$[r]$ are at most those of ForkAES$[r-1]$.

**Statement (2):** The security guarantees of ForkAES$[r']$ are at most those of a variant of $10 - (r' - r)$ rounds of ForkAES$[r]$, for all $r, r' \leq 10$.

So, our attacks on nine-round ForkAES almost directly carry over to attacks on full-round ForkAES[6], only the indices have to be shifted. Moreover, reduced-round attacks are applicable to all variants ForkAES[7] through ForkAES[10].