ForkAES: a Tweakable Forkcipher

Elena Andreeva[1], Reza Reyhanitabar[2], Kerem Varici[1] and Damian Vizár[3]

[1] imec-COSIC, KU Leuven, Belgium
elena.andreeva@esat.kuleuven.be, kerem.varici@esat.kuleuven.be
[2] Elektrobit Automotive GmbH, Germany
reza.reyhanitabar@elektrobit.com
[3] CSEM, Switzerland
damian.vizar@csem.ch

Abstract. In the recent post-CAESAR era, it became clear that authenticated encryption optimized for *short* messages is a research problem that is both highly relevant, and not yet fully solved. The concept of forkcipher, a new kind of cryptographic primitive, has been proposed as a mean to sidestep the overcome the limitations of AE schemes based on typical primitives (such as blockciphers). This paper introduces a forkcipher construction that is based on the tweakable blockcipher KIASU, which is in turn based on AES.

Keywords: Authenticated encryption, short messages, lightweight cryptography, forkcipher, ForkAES.

1 Introduction

A forkcipher is tweakable symmetric cryptographic primitive with a fixed input length and a fixed output length that is *expanding* (i.e., it output more bits than it takes as input). It has been reently proposed by Andreeva et al., primarily as a mean to construct authenticated encryption (AE) that is higly efficient for the shortest messages [1, 2]. Andreeva et al. formalized the syntax and security notion of tweakable forkciphers, proposed a framework that lifts an iterated tweakable blockciper to a forkcipher, designed ForkSKINNY (a forkcipher construction based on SKINNY tweakable blockcipher [4]), and introduced provably secure forkcipher modes for AE efficient for the shortest queries.

This paper describes ForkAES, a preliminary instance of a forkcipher. ForkAES uses a vanilla version of the framework by Andreeva et al. to transform the tweakable blockcipher KIASU [13] into an iterated forkcipher. KIASU itself is a tweakable blockcipher that is based on AES [7] and the Tweakey framework [14].

Recent cryptanalysis results [3] have evidenced that further strnegthening of this ForkAES can benefit its security.

2 Preliminaries

All strings are binary strings. The set of all strings of length n (for a positive integer n) is denoted $\{0,1\}^n$. We let $\{0,1\}^{\leq n}$ denote the set of all strings of length at most n. We denote by $\mathrm{Perm}(n)$ the set of all permutations of $\{0,1\}^n$.

For a string X of ℓ bits, we let $X[i]$ denote the i^{th} bit of X for $i = 0, \ldots, \ell-1$ (starting from the left) and $X[i \ldots j] = X[i] \| X[i+1] \| \ldots \| X[j]$ for $0 \leq i < j < \ell$. We let $\mathsf{left}_\ell(X) = X[0 \ldots (\ell-1)]$ denote the ℓ leftmost bits of X and $\mathsf{right}_r(X) = X[(|X|-r) \ldots (|X|-1)]$ the r rightmost bits of X, such that $X = \mathsf{left}_\chi(X) \| \mathsf{right}_{|X|-\chi}(X)$

for any $0 \leq \chi \leq |X|$. We let $(L, R) = \mathsf{lsplit}_{X,n}$ denote splitting a string $X \in \{0,1\}^*$ into two parts such that $L = \mathsf{left}_{\min(|X|,n)}(X)$ and $R = \mathsf{right}_{|X|-|L|}(X)$. In particular, for $n \geq |X|$ we have $(X, \varepsilon) = \mathsf{lsplit}_{X,n}$. We further let $(M', M_*) = \mathsf{msplit}_n(M)$ denote a splitting of a string $M \in bits*$ into two parts $M' \| M_* = M$, such that $|M_*| \equiv |M| \pmod n$, and $0 \leq |M_*| \leq n$, where $|M_*| = 0$ if and only if $|M| = 0$. We let $(C', C_*, T) = \mathsf{csplit}_n(C)$ splitting a string C of at least n bits into three parts $C' \| C_* \| T = C$, such that $|C_*| = n$, $|T| \equiv |C| \pmod n$, and $0 \leq |T| \leq n$, where $|T| = 0$ if and only if $|C| = n$. Finally, we let $C'_1, \ldots, C'_m, C_*, T \leftarrow \mathsf{csplit\text{-}b}_n(C)$ denote a version of $\mathsf{csplit}_n(C)$, where the string C' further gets partitioned into $|C'|_n$ blocks of n bits, such that $C' = C'_1 \| \ldots \| C'_m$.

Given a string X and an integer n, we let $X_1, \ldots, X_x, X_* \xleftarrow{n} X$ denote partitioning X into n-bit blocks, such that $|X_i| = n$ for $i = 1, \ldots, x$, $0 \leq |X_*| \leq n$ and $X = X_1 \| \ldots \| X_x \| X_*$, so $x = \max(0, \lfloor X/n \rfloor - 1)$. We let $|X|_n = \lceil X/n \rceil$. Given a (possibly implicit) positive integer n and an $X \in \{0,1\}^*$, we let $X \| 10^*$ denote $X \| 10^{n-(|X| \bmod n)-1}$ for simplicity.

The symbol $\perp$ denotes an error signal, or an undefined value. We denote by $X \leftarrow\!\!\$\ \mathcal{X}$ sampling an element X from a finite set $\mathcal{X}$ following the uniform distribution.

3 Forkcipher

In this section, we briefly state the syntax and security goals of a forkcipher. We note that the formalism differs from that of Andreeva et al. [1,2] in that the decryption and reconstruction algorithm are merged into a multipurpose "decryption" in the latter work.

Syntax. A forkcipher is a triple of deterministic algorithms, the encryption algorithm $\mathsf{F} : \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^{2n}$, the inversion algorithm $\mathsf{F}^{-1}\{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \times \{0,1\} \to \{0,1\}^n$ and the tag reconstruction algorithm $\mathsf{F}^\rho\{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \times \{0,1\} \to \{0,1\}^n$. We call k, n and $\mathcal{T}$ the keysize, blocksize and tweak space of F, respectively.

A tweakable forkcipher F meets the *correctness condition*, if for every $K, T, M, \beta \in \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \times \{0,1\}$ we have

$$\mathsf{F}^{-1}(K, T, \mathsf{F}(K, T, M)[(\beta \cdot n) \ldots (\beta \cdot n + n - 1)], \beta) = M$$

and

$$\mathsf{F}(K, T, M)[((1-\beta)\cdot n) \ldots ((1-\beta)\cdot n + n - 1)] = \mathsf{F}^\rho(K, T, \mathsf{F}(K, T, M)[(\beta \cdot n) \ldots (\beta \cdot n + n - 1)], \beta).$$

Security Definition. We define the security of forkciphers by an indistiguishability experiment based on the security games in Figure 1.

An adversary $\mathcal{A}$ that aims at breaking a tweakable forkcipher F plays the games **prtfp-real** and **prtfp-ideal** and define the advantage of $\mathcal{A}$ at distinguishing F from a random tweakable injection in a *chosen ciphertext attack* as

$$\mathbf{Adv}^{\mathrm{prtfp}}_{\mathsf{F}}(\mathcal{A}) = \Pr[\mathcal{A}^{\mathbf{prtfp\text{-}real}_{\mathsf{F}}} \Rightarrow 1] - \Pr[\mathcal{A}^{\mathbf{prtfp\text{-}ideal}_{\mathsf{F}}} \Rightarrow 1].$$

4 ForkAES

We design ForkAES, a forkcipher construction. It is best described by its name: it is an AES-based design with its internal state *forked* after half of the rounds to produce *two* redundant 128-bit output blocks. We also add a tweak to facilitate the design of simple modes of operation. ForkAES is obtained by combining two ingredients: the KIASU [13,14] tweakable blockcipher (which is, in turn, a derivative of AES, hence the name), and the *iterate-fork-iterate* paradigm [1,2].

Game **prtfp-real**$_{\mathsf{F}}$

$K \leftarrow\!\!\$\ \{0,1\}^k$
$b \leftarrow \mathcal{A}^{\text{Enc},\text{Dec}}$
return b

Oracle $\text{Enc}(T, M)$
return $\mathsf{F}(K, T, M)$

Oracle $\text{Dec}(T, C, \beta)$
return $\mathsf{F}^{-1}(K, T, C, \beta)$

Game **prtfp-ideal**$_{\mathsf{F}}$

for $T \in \mathcal{T}$ **do** $\pi_{T,0}, \pi_{T,1} \leftarrow\!\!\$\ \text{Perm}(n)$
$b \leftarrow \mathcal{A}^{\text{Enc},\text{Dec}}$
return b

Oracle $\text{Enc}(T, M)$
return $\pi_{T,0}(M) \| \pi_{T,1}(M)$

Oracle $\text{Dec}(T, C, \beta)$
return $\pi^{-1}_{T,\beta}(C)$

Figure 1: Games **prtfp-real**, and **prtfp-ideal** used to define security of a (strong) forkcipher.

4.1 Specification

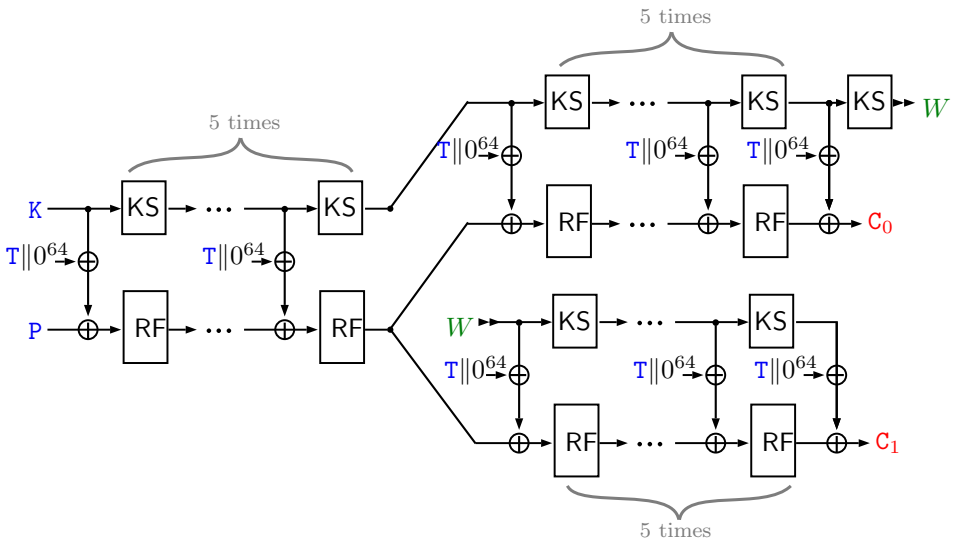

Figure 2: Illustration of an encryption by ForkAES. A 128 bit plaintext P, a 128 bit key K and 64 bit tweak T (all in blue) are used to compute a 256 bit ciphertext $\mathtt{C} = \mathtt{C}_0 \| \mathtt{C}_1$ (in red). RF denotes a single iteration of the AES round function and KS denotes a single iteration of the AES keyschedule.

ForkAES is a deterministic cryptographic algorithm which takes a 128-bit plaintext P, a 64-bit tweak T and a 128-bit secret key K as input, and outputs a 256-bit ciphertext C (i.e., $\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P}) = \mathtt{C}$).

It is based on the tweakable blockcipher KIASU. In KIASU, a round function based on the SubBytes, Shiftrows and Mixcolumn operations of AES is iteratively applied to the plaintext block. Following the TWEAKEY framework [14], the secret key and tweak are used to generate subkeys which are xored to the intermediate internal state before every

application of the round function.

Iterate-fork-iterate. How ForkAES differs from both AES and KIASU is that after half of the rounds, the encryption is forked and two copies of the internal states are further processed with different sets of *independent* subkeys. The additional required subkeys are generated by doing the necessary number of extra iterations of the key schedule (beyond what would have been done in the original (tweakable) blockcipher).

Inverse algorithms. Associated to ForkAES are the decryption algorithm $\mathsf{ForkAES}^{-1}$ and the *reconstruction* algorithm $\mathsf{ForkAES}^{\rho}$. Because the two output blocks produced by ForkAES are redundant, either one of them is sufficient for decryption. The decryption algorithm thus takes a secret key $\mathtt{K}$, a tweak $\mathtt{T}$, a half-ciphertext $\mathtt{C}$ of 128 bits, and a bit b that indicates whether this is the left half or the right half, and inverts the "fork" indicated by b and then the initial common processing. For every $\mathtt{K}, \mathtt{P} \in \{0,1\}^{128}$ and every $\mathtt{T} \in \{0,1\}^{64}$ we have

$$\mathtt{P} = \mathsf{ForkAES}^{-1}\left(\mathtt{K}, \mathtt{T}, \mathsf{left}_n(\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P})), 0\right) = \mathsf{ForkAES}^{-1}\left(\mathtt{K}, \mathtt{T}, \mathsf{right}_n(\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P})), 1\right).$$

Similarly, the redundancy can be used to recompute one output block from the other which is what the reconstruction algorithm does. It takes a secret key $\mathtt{K}$, a tweak $\mathtt{T}$, a half-ciphertext $\mathtt{C}$ of 128 bits, and a bit b that indicates whether this is the left half or the right half, inverts the indicated "fork", and then recomputes the other one. For every $\mathtt{K}, \mathtt{P} \in \{0,1\}^{128}$ and every $\mathtt{T} \in \{0,1\}^{64}$ we have

$$\mathsf{ForkAES}^{\rho}\left(\mathtt{K}, \mathtt{T}, \mathsf{left}_n(\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P})), 0\right) = \mathsf{right}_n(\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P}))$$

and

$$\mathsf{ForkAES}^{\rho}\left(\mathtt{K}, \mathtt{T}, \mathsf{right}_n(\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P})), 1\right) = \mathsf{left}_n(\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P}))$$

The formal algorithmic description of all three algorithms is given in Figure 3, and the encryption operation is illustrated in Figure 2.

To generate the round keys, we set the secret key as the first round key, iterate the key schedule of AES 16 times, and xor the tweak to the 8 leftmost bytes of each round key. This is exactly what is done in KIASU, except we iterate the key schedule 6 more times. The round key generation algorithm is described in Figure 3.

4.2 Security Evaluation

In this section, we briefly discuss the security of ForkAES against the most important cryptanalytic attacks. We only consider classical black-box attacks, i.e., we do not consider side-channel attacks.

Differential Cryptanalysis. Differential crpyptanalysis is one of the most powerful security analysis methods and showing the security of a cipher against it is essential part of the security evaluation. For a cipher based on the Substitution Permutation Network (SPN) the analysis is relatively easy and well-understood and it is based on counting the number of active s-boxes over the cipher rounds. When the active s-boxes reach a certain threshold then the cipher is assumed to be secure against differential cryptanalysis. For example, in the case of AES in the single-key model, one can guarantee at least 25 active s-boxes for a differential path of four rounds due to the careful choice of a permutation layer (which is a diffusion matrix with branching number five). If each active s-box reaches the maximal differential probability of the AES S-box $p_{max} = 2^{-6}$, then the probability of the differential path becomes $2^{-150} < 2^{-128}$. Hence, four AES rounds already provide enough protection. Since our ForkAES design uses the AES round function, we can easily deduce that our design will provide enough security in this setting after four rounds against differential attacks in the single-key model.

Algorithm $\mathsf{ForkAES}(\mathtt{K}, \mathtt{T}, \mathtt{P})$
  $K_0, \ldots, K_{16} \leftarrow \mathsf{KeySched}(\mathtt{K}, \mathtt{T})$
  $S \leftarrow \mathtt{P}$
  for $i \leftarrow 0$ **to** 4 **do**
    $S \leftarrow S \oplus K_i$
    $S \leftarrow \mathsf{AESrnd}(S)$
  end for
  $S_0 \leftarrow S$; $S_1 \leftarrow S$
  for $i \leftarrow 5$ **to** 9 **do**
    $S_0 \leftarrow S_0 \oplus K_i$
    $S_0 \leftarrow \mathsf{AESrnd}(S_0)$
  end for
  $\mathtt{C}_0 \leftarrow S_0 \oplus K_{10}$
  for $i \leftarrow 11$ **to** 15 **do**
    $S_1 \leftarrow S_1 \oplus K_i$
    $S_1 \leftarrow \mathsf{AESrnd}(S_1)$
  end for
  $\mathtt{C}_1 \leftarrow S_1 \oplus K_{16}$
  return $\mathtt{C}_0 \| \mathtt{C}_1$
end Algorithm

Algorithm $\mathsf{ForkAES}^{-1}(\mathtt{K}, \mathtt{T}, \mathtt{C}, b)$
  $K_0, \ldots, K_{16} \leftarrow \mathsf{KeySched}(\mathtt{K}, \mathtt{T})$
  $S \leftarrow \mathtt{C} \oplus K_{10+b\cdot 6}$
  for $i \leftarrow 9 + b \cdot 6$ **to** $5 + b \cdot 6$ **do**
    $S \leftarrow \mathsf{AESrnd}^{-1}(S)$
    $S \leftarrow S \oplus K_i$
  end for
  for $i \leftarrow 4$ **to** 0 **do**
    $S \leftarrow \mathsf{AESrnd}^{-1}(S)$
    $S \leftarrow S \oplus K_i$
  end for
  return S
end Algorithm

Algorithm $\mathsf{ForkAES}^{\rho}(\mathtt{K}, \mathtt{T}, \mathtt{C}, b)$
  $K_0, \ldots, K_{16} \leftarrow \mathsf{KeySched}(\mathtt{K}, \mathtt{T})$
  $b' \leftarrow b \oplus 1$
  $S \leftarrow \mathtt{C} \oplus K_{10+b\cdot 6}$
  for $i \leftarrow 9 + b \cdot 6$ **to** $5 + b \cdot 6$ **do**
    $S \leftarrow \mathsf{AESrnd}^{-1}(S)$
    $S \leftarrow S \oplus K_i$
  end for
  for $i \leftarrow 5 + b' \cdot 6$ **to** $9 + b' \cdot 6$ **do**
    $S \leftarrow S \oplus K_i$
    $S \leftarrow \mathsf{AESrnd}(S)$
  end for
  $\mathtt{C}' \leftarrow S \oplus K_{10+b'\cdot 6}$
  return $\mathtt{C}'$
end Algorithm

Algorithm $\mathsf{KeySched}(\mathtt{K}, \mathtt{T})$
  $K_0 \leftarrow \mathtt{K} \oplus \mathsf{Rwfy}(\mathtt{T} \| 0^{64})$
  $W_0, \ldots, W_3 \xleftarrow{32} \mathtt{K}$
  for $i \leftarrow 1$ **to** 16 **do**
    $\mathsf{tmp} \leftarrow \mathsf{RotWord}(\mathsf{W[3]})$
    $W'_0 \leftarrow W_0 \oplus \mathsf{SubWord}(\mathsf{tmp}) \oplus \mathsf{Rcon}[i]$
    for $j \leftarrow 1$ **to** 3 **do**
      $W'_j \leftarrow W_j \oplus W'_{j-1}$
    end for
    for $j \leftarrow 0$ **to** 3 **do**
      $W_j \leftarrow W'_j$
    end for
    $K_i \leftarrow W_0 \| W_1 \| W_2 \| W_3 \oplus \mathsf{Rwfy}(\mathtt{T} \| 0^{64})$
  end for
  return $K_0, \ldots, K_{16}$
end Algorithm

Algorithm $\mathsf{AESrnd}(S)$
  $S \leftarrow \mathsf{SubBytes}(S)$
  $S \leftarrow \mathsf{ShifRows}(S)$
  $S \leftarrow \mathsf{MixColumns}(S)$
  return S
end Algorithm

Algorithm $\mathsf{AESrnd}^{-1}(S)$
  $S \leftarrow \mathsf{iMixColumns}(S)$
  $S \leftarrow \mathsf{iShifRows}(S)$
  $S \leftarrow \mathsf{iSubBytes}(S)$
  return S
end Algorithm

Figure 3: The algorithms $\mathsf{ForkAES}$, $\mathsf{ForkAES}^{-1}$ and $\mathsf{ForkAES}^{\rho}$. The function $Y = \mathsf{Rwfy}(X)$ (from "rowify") is a byte-transposition of a 128-bit string X that maps $Y_i = X_{4*(i \bmod 4)+\lfloor i/4 \rfloor}$.

Table 1: Upper bounds on probabilities of related-TWEAKEY differential characteristics [13, Table 4.1].

Rounds	Active S-boxes	Probability (upper bound)	Method
1	0	2^0	trivial
2	0	2^0	trivial
3	1	2^{-6}	Matsui's
4	8	2^{-48}	Matsui's
5	≥ 14	2^{-84}	Matsui's
7	≥ 22	2^{-132}	extended split $(3R + 4R)$

Related-TWEAKEY Attacks. The extra freedom provided from key K (in our case tweak T as well) makes the security evaluation of ciphers against related-key (in our case related-tweakey) attacks more challenging. Over the years, many search algorithms [5,6,11,12,15,16] were given to compute an upper bound for the related-key differential characteristics. The KIASU designers gave a comprehensive related-key analysis for KIASU by extending the search algorithms to cover the related-tweak option and we summarize their results in Table 1. Our design is based on the KIASU algorithm and its tweakey schedule and thus a closer inspection reveals that the latter results also apply to ForkAES.

Meet-in-the-Middle Attack. There are numerous meet-in-the-middle attacks performed against AES [8–10] and for all those attacks the key schedule plays an important role. In these attacks partial encryption/decryption is done by guessing keys to prepare pre-computed tables. To reduce the amount of guessed key bytes (and respective attack complexities), the existing linear relations of the AES key schedule are exploited. In our design, we use KIASU as our core encryption operation which in turn replies on the AES cipher with the tweak addition to key schedule. The tweak is a fixed and known constant value T and therefore the existing meet-in-the-middle attacks for AES-128 will apply to both KIASU and our design.

Security Against Other Attacks. Our forkcipher ForkAES is based purposely on the AES block cipher regarding round function and key schedule designs. Moreover, we borrow the KIASU tweak (tweak and key) treatment to support the use of the additional tweak input in our design. Since we do not introduce any novel design complexities, the security of our forkcipher design can be reduced to the security of the AES and KIASU ciphers for further type of attacks.

Third party cryptanalysis. An independent cryptanalysis of ForkAES by Banik et al. showed, that a round-reduced version with 9 rounds (instead of full 10 rounds) can be attacked with practical complexity [3]. More precisely, Banik et al. showed that differential, impossible-differential and yoyo attacks exist that exploit the reconstruction interface, and that further rectangle and impossible-differential attacks exist that only use the encryption queries. The implications of these attacks were exemplified by transforming them into forgery attacks against the modes presented by Andreeva et al.

5 Discussion

This brief paper presented ForkAES, a preliminary construction of the forkcipher primitive. While it demonstrates the performance advantages that can be achieved through the iterate-fork-iterate paradigm, it is also an immature construction; while no efficient attack

on the full version is known, its security margin is certainly insufficient. We refer the reader looking for a secure forkcipher instance to the work that introduces ForkSKINNY [1, 2].

Acknowledgments.

Elena Andreeva was supported in part by the Research Council KU Leuven C1 on Security and Privacy for Cyber-Physical Systems and the Internet of Things with contract number C16/15/058 and by the Research Council KU Leuven, C16/18/004, through the EIT Health RAMSES project, through the IF/C1 on New Block Cipher Structures, and through the NIST project. In addition, this work was supported by the European Commission through the Horizon 2020 research and innovation programme under grant agreement H2020-DS-2014-653497 PANORAMIX and through the grant H2020-DS-SC7-2016-740507 Eunity. The work is supported in part by funding from imec of the Flemish Government. This work was also partly supported by the EU H2020 POMEGRANATE project, funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 708815, while Reza Reyhanitabar was a Marie Skłodowska-Curie Individual Fellow in KU Leuven.

References

[1] Andreeva, E., Lallemand, V., Purnal, A., Reyhanitabar, R., Roy, A., Vizár, D.: Forkcipher: a New Primitive for Authenticated Encryption of Very Short Messages. In: ASIACRYPT 2019. LNCS, Springer (2019)

[2] Andreeva, E., Lallemand, V., Purnal, A., Reyhanitabar, R., Roy, A., Vizár, D.: Forkcipher: a new primitive for authenticated encryption of very short messages. Cryptology ePrint Archive, Report 2019/1004 (2019), https://eprint.iacr.org/2019/1004

[3] Banik, S., Bossert, J., Jana, A., List, E., Lucks, S., Meier, W., Rahman, M., Saha, D., Sasaki, Y.: Cryptanalysis of forkaes. In: Deng, R.H., Gauthier-Umaña, V., Ochoa, M., Yung, M. (eds.) Applied Cryptography and Network Security - 17th International Conference, ACNS 2019, Bogota, Colombia, June 5-7, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11464, pp. 43–63. Springer (2019), https://doi.org/10.1007/978-3-030-21568-2_3

[4] Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9815, pp. 123–153. Springer (2016), https://doi.org/10.1007/978-3-662-53008-5_5

[5] Biryukov, A., Nikolic, I.: Automatic search for related-key differential characteristics in byte-oriented block ciphers: Application to aes, camellia, khazad and others. In: Gilbert, H. (ed.) Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6110, pp. 322–344. Springer (2010), https://doi.org/10.1007/978-3-642-13190-5_17

[6] Biryukov, A., Nikolic, I.: Search for related-key differential characteristics in DES-Like ciphers. In: Joux, A. (ed.) Fast Software Encryption - 18th International Workshop,

FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6733, pp. 18–34. Springer (2011), https://doi.org/10.1007/978-3-642-21702-9_2

[7] Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography, Springer (2002), https://doi.org/10.1007/978-3-662-04722-4

[8] Demirci, H., Selçuk, A.A.: A meet-in-the-middle attack on 8-round AES. In: Nyberg, K. (ed.) Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers. Lecture Notes in Computer Science, vol. 5086, pp. 116–126. Springer (2008), https://doi.org/10.1007/978-3-540-71039-4_7

[9] Derbez, P., Fouque, P., Jean, J.: Improved key recovery attacks on reduced-round AES in the single-key setting. In: Johansson, T., Nguyen, P.Q. (eds.) Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7881, pp. 371–387. Springer (2013), https://doi.org/10.1007/978-3-642-38348-9_23

[10] Dunkelman, O., Keller, N., Shamir, A.: Improved single-key attacks on 8-round AES-192 and AES-256. In: Abe, M. (ed.) Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6477, pp. 158–176. Springer (2010), https://doi.org/10.1007/978-3-642-17373-8_10

[11] Emami, S., Ling, S., Nikolić, I., Pieprzyk, J., Wang, H.: The resistance of present-80 against related-key differential attacks. Cryptography Commun. 6(3), 171–187 (Sep 2014), http://dx.doi.org/10.1007/s12095-013-0096-8

[12] Fouque, P., Jean, J., Peyrin, T.: Structural evaluation of AES and chosen-key distinguisher of 9-round AES-128. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. Lecture Notes in Computer Science, vol. 8042, pp. 183–203. Springer (2013), https://doi.org/10.1007/978-3-642-40041-4_11

[13] J. Jean, I. Nikolić, T.P.: KIASU v1 (2014), "https://competitions.cr.yp.to/round1/kiasuv1.pdf"

[14] Jean, J., Nikolic, I., Peyrin, T.: Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 274–288. Springer (2014)

[15] Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Wu, C., Yung, M., Lin, D. (eds.) Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers. Lecture Notes in Computer Science, vol. 7537, pp. 57–76. Springer (2011), https://doi.org/10.1007/978-3-642-34704-7_5

[16] Siwei Sun, Lei Hu, P.W.K.Q.X.M.L.S.: Automatic security evaluation and (related-key) differential characteristic search: Application to simon, present, lblock, des(l) and other bit-oriented block ciphers. Cryptology ePrint Archive, Report 2013/676 (2013), https://eprint.iacr.org/2013/676