Masked Iterate-Fork-Iterate: A New Design Paradigm for Tweakable Expanding Pseudorandom Function

Elena Andreeva, Benoît Cogliati, Virginie Lallemand, Marine Minier, Antoon Purnal, Arnab Roy

Masked Iterate-Fork-Iterate: A new Design Paradigm for Tweakable Expanding Pseudorandom Function

Elena Andreeva[1], Benoit Cogliati[2], Virginie Lallemand[3], Marine Minier[3], Antoon Purnal[4], and Arnab Roy[5]

[1] TU Wien, Vienna, Austria elena.andreeva@tuwien.ac.at
[2] Thales DIS France SAS, Meudon, France benoit.cogliati@gmail.com
[3] Université de Lorraine, CNRS, Inria, LORIA, Nancy, France firstname.name@loria.fr
[4] KU Leuven, Leuven, Belgium antoon.purnal@kuleuven.be
[5] University of Klagenfurt, Klagenfurt, Austria arnab.roy@aau.at

Abstract. Many modes of operations for block ciphers or tweakable block ciphers do not require invertibility from their underlying primitive. In this work, we study fixed-length Tweakable Pseudorandom Function (TPRF) with large domain extension, a *novel primitive* that can bring high security and significant performance optimizations in symmetric schemes, such as (authenticated) encryption.

Our first contribution is to introduce a new design paradigm, derived from the Iterate-Fork-Iterate construction, in order to build n-to-αn-bit ($\alpha \geq 2$), *n-bit secure*, domain expanding TPRF. We dub this new generic composition masked Iterate-Fork-Iterate mIFI. We then propose a concrete TPRF instantiation ButterKnife that expands an n-bit input to $8n$-bit output via a public tweak and secret key. ButterKnife is built with high efficiency and security in mind. It is fully parallelizable and based on Deoxys-BC, the AES-based tweakable block cipher used in the authenticated encryption winner algorithm in the defense-in-depth category of the recent CAESAR competition. We analyze the resistance of ButterKnife to differential, linear, meet-in-the-middle, impossible differentials and rectangle attacks. A special care is taken to the attack scenarios made possible by the multiple branches.

Our next contribution is to design and provably analyze two new TPRF-based deterministic authenticated encryption (DAE) schemes called SAFE and ZAFE that are highly efficient, parallelizable, and offer $(n+\min(n,t))/2$ bits of security, where n, t denote respectively the input block and the tweak sizes of the underlying primitives. We further implement SAFE with ButterKnife to show that it achieves an encryption performance of 1.06 c/B for long messages on Skylake, which is $33-38\%$ faster than the comparable Crypto'17 TBC-based ZAE DAE. Our second candidate ZAFE, which uses the same authentication pass as ZAE, is estimated to offer a similar level of speedup. Besides, we show that ButterKnife, when used in Counter Mode, is *slightly faster* than AES (0.50 c/B vs 0.56 c/B on Skylake).

Keywords: tweakable pseudorandom functions, expanding primitives, deterministic authenticated encryption, beyond-birthday-bound security

1 Introduction

Building blocks. Block ciphers (BCs) are fundamental primitives in symmetric cryptography. AES [AES01] is the most popular block cipher in use today, a fact that has prompted processor vendors, like Intel and ARM, to equip their products with AES hardware acceleration, enabling excellent software performance.

In [LRW02], Liskov *et al.* proposed Tweakable block ciphers (TBC) as an extension of classical block ciphers by adding a public tweak input. Secure TBCs, similarly to secure BCs as pseudorandom permutations (PRPs), are modeled as tweak-keyed pseudorandom permutations (TPRPs). TBCs allow for building iterative symmetric schemes of higher security [IMPS17, LN17, PS16] than their BC-based counterparts, thanks to their increased input size. [LRW02] provided the security analysis of a TBC built from a secure BC, where the tweak size is either same as the block size or a longer tweak (than block size) that is hashed to the block size. Successive results investigated how to securely integrate a tweak to a BC. A number of recent TBC designs follow the so-called TWEAKEY [JNP14] design framework. These include SKINNY [BJK+16], and the AES-based Deoxys-BC, Joltic-BC and KIASU-BC.

Forkciphers [ALP+19] (FC) and their multi-forkcipher [ABPV21] (MFC) generalization are recently proposed tweakable symmetric primitives. MFCs are fixed-length n-to-αn ($\alpha = 2$ for forkciphers) domain-extending functions that come with forward, inverse, and a reconstruction evaluation functionalities. To date, ForkSkinny [ALP+19] is the only secure forkcipher instance. Each ForkSkinny branch can be viewed as an instance of the SKINNY TBC. Since each branch forms a permutation under a fixed key, the maximal security is set by the birthday bound to $n/2$.

Because the $2n$-bit domain extension is realized cheaper than 2 SKINNY calls, ForkSkinny achieves performance improvements in (r)PAEF authenticated encryption (AE) [ALP+19] modes, and in counter-style (CTR) encryption modes [ABPV21]. Yet, like with classical BC-based CTR-style encryption, the latter modes require a primitive that makes *forward-only evaluations*.

Tweakable pseudorandom function. A more natural candidate primitive for the latter optimizations hence is a novel $2n$-to-αn pseudorandom function (PRF), which can achieve n-bit (rather than $n/2$-bit) security thanks to its $2n$-bit input size. Since a tweakable PRF (TPRF) is equivalent to a PRF with a bigger input space (that subsumes the tweak), the only point of introducing a tweak/block distinction would be to bring down the computational cost of processing changes of these different input parts. As most dedicated PRFs in symmetric cryptography are usually build upon existing block ciphers [BKR98, CS16, MN17a, MN17b], a TPRF then becomes a preferred building block here

as it can be built upon existing TBCs that readily support the increased $2n$-bit input space via their tweak and block as compared to $2n$-bit BCs building blocks.

Beyond birthday bound (BBB). BBB security, or security higher than $n/2$-bit and as close as possible to n-bit is required nowadays for many cryptographic applications in the face of the present advancements in computation, concentration of resources in powerful entities, or multi-user (device) security aspects.

Fixed-length n-to-αn-bit TPRF primitives have not been researched to date. In this work we set to answer the main question: "*Can we design an efficient, provably and cryptanalytically sound, n-bit secure fixed-length n-to-αn TPRF for $\alpha \geq 2$ and show highly efficient, beyond birthday bound secure applications of it in symmetric cryptography?*"

In this work we focus on authenticated encryption (AE) applications of TPRF and in particular, we target secure deterministic AE as it offers high security guarantees.

Authenticated encryption. AE is the symmetric standard for guaranteeing *both* privacy and authenticity for the bulk of communication and data at rest nowadays. Contemporary AE schemes use a *nonce*. Some of the most popular nonce-based AE examples are OCB [RBBK01] and the NIST-recommended [NIS07] modes GCM [MV04] and CCM [Hou05]. Nonces deter ciphertext repetitions when identical messages are processed. One of the main adoption reasons for nonce-based AE is that a nonce can be implemented as a random value or a synchronized state updated with every message. Oftentimes, however, when the device is equipped with a weak or flawed software randomness source, or has a restricted secure storage for its state, the nonce may repeat. Users can also accidentally mishandle nonces or set them to constants. The consequence of nonce repetitions in counter (CTR) mode-style AE (e.g. OCB) is a loss in confidentiality. Recent nonce-misuse attacks illustrate the severity of nonce repetitions in *practice*. In 2016 Böck *et al.* [BZD+16] showed that the authenticity of AES-GCM in TLS can be completely broken where servers repeated the nonce. They also showed that weakness can serve further for successful injections of valid content into encrypted sessions. Vanhoef and Piesens [VP] introduced the key reinstallation attack which forces nonce repetitions and breaks the WPA2 wireless protocol.

Deterministic authenticated encryption (DAE). In some applications, the ciphertext differentiation (of repeated messages) is not a desirable feature. In that case the AE deterministic feature of the algorithm might be exactly what the application requires, such as, for example, access to encrypted database storage. In such scenarios it is better to use nonce-misuse (NM) resistant AE [RS06]. Deterministic AE (DAE) schemes are nonce misuse-resistant in the sense of [RS06]. Secure DAE leak only the repetition of repeated messages, but they retain security over distinct messages even in the face of nonce repetitions. DAE schemes achieve the highest AE security levels and are inherently two-pass designs. In a nutshell, DAEs provide security independently of the (mis)use of nonces.

A DAE scheme deterministically transforms a key K, associated data A, and a message M into a ciphertext C. SIV [RS06] is one of the most well-

known DAE. The GCM-SIV was proposed by Gueron and Lindell [GL15] as an instantiation of SIV. A slightly modified version AES-GCM-SIV was later defined in RFC 8452 [GLL19]. DAE also has its merits in the context of lightweight AE [BBLT18] when devices lack a secure randomness generator or memory.

In [IM16] Iwata and Minematsu showed that the GCM-SIV security bound of $\frac{q}{2^{(n-k)/2}}$, where q is the number of queries, and 2^k denotes the maximum block length of all the encryption and decryption queries, is tightly matched by a trivial distinguishing attack in just 2^{48} queries of one message block for $n = 128$ and $k = 32$. The next year Gueron *et al.* [GLL17] proposed AES-GCM-SIV instantiation with a claimed security bound of $\frac{QR^2}{2^{n-k}}$ with Q being the number of distinct nonces, R the maximal number of repetitions of any nonce both over encryption queries, and the maximum message length 2^{k-1} blocks. A year later Iwata and Seurin [IS17] discovered a flaw in the proof of the (AES)-GCM-SIV to show that the offered security is actually worse than assumed, namely $\frac{QR^2}{2^{n-2k}}$, or further lower than the birthday bound. As such (AES)-GCM-SIV, although performing close to only 1 cpb (despite its two-pass structure), failed to deliver strong security guarantees that hardly match the birthday bound security.

To address these security concerns several recent DAE designs have targeted *beyond birthday bound security*. In [IM16] Iwata and Minematsu propose GCM-SIV2 which is secure up to about $2^{2n/3}$ queries. In [PS16] Peyrin and Seurin presented a nonce-based AE scheme called Synthetic Counter in Tweak (SCT) that is based on a TBC. SCT achieves BBB security when nonces do not repeat but its security degrades to the birthday bound with the reuse of nonces. In [IMPS17] Iwata *et al.* propose the ZAE DAE mode, which achieves BBB security and processes $n(n+t)/(2n+t)$ input bits per TBC. The performance result was possible due to the use of the ZMAC authentication that "absorbs" $(n+t)$ bits per authenticated block. ZAE is instantiated with the Deoxys-BC [JNPS18] and SKINNY [BJK+16] TBCs. Table 1 summarizes the security and performance of these schemes.

Table 1: Security against nonce respecting (NR) and nonce-misuse (NM) adversaries and performance comparison of DAE schemes for long messages on Intel Skylake. By n we denote the block size, t the tweak size, Q the number of distinct nonces (in enc.), R is the maximal number of nonce repetitions (in enc.), σ the total length in blocks of queried messages, q the number of encryption queries, and the maximum message length by 2^{k-1} blocks.

AE Scheme	Security (NR)	Security (NM)	Performance [c/B]
AES-GCM-SIV [GLL17]	$\frac{Q}{2^{n-2k}}$	$\frac{QR^2}{2^{n-2k}}$	0.83
SCT [PS16]	$\frac{q}{2^n} + \frac{\sigma^2}{2^{n+t}}$	$\frac{q^2}{2^n} + \frac{R\sigma}{2^t}$	1.74
ZAE [IMPS17]	$\frac{\sigma^2}{2^{n+\min(n,t)}}$	$\frac{\sigma^2}{2^{n+\min(n,t)}}$	1.46
SAFE; ZAFE [This work]	$\frac{\sigma^2}{2^{n+\min(n,t)}}$	$\frac{\sigma^2}{2^{n+\min(n,t)}}$	1.06; 1.10

Security-wise the ZAE DAE excels over its predecessors. Efficiency-wise the AES-GCM-SIV still significantly outperforms ZAE (see detailed performance numbers in Table 5). One reason is that using TBCs to attain BBB secure DAE schemes comes with some performance penalty when compared directly to BC-based designs, such as the GCM-SIV DAE, admittedly with lower security margin, due to the computational overhead associated with the tweakey processing.

Our next research question in this work is: "*Is it possible to use a fixed-length TPRF to design a DAE scheme with security guarantees comparable to* ZAE *but with an improved performance much closer to* GCM-SIV*?*"

1.1 Contributions

Novel, generic n-to-αn-bit TPRF. We propose a generic masked Iterate-Fork-Iterate mIFI method for building an n-bit secure TPRF. mIFI uses $(\alpha + 1)$ independent permutations, the first one to generate a fork state and the rest for the output branches. We apply the internal fork state to mask the outputs of each branch to preclude invertibility and obtain as a result a secure TPRF.

More concretely, we build a TPRF with fixed inputs a key K, a tweak T, an n-bit message M and which outputs a fixed-length ciphertext C of αn bits via $(\alpha + 1)$ calls to the underlying independent random permutations where $\alpha \geq 2$. While the design bears similarities with the iterate-fork-iterate [ALP+19] forkcipher generic composition, the proof requires an entirely different approach that is enabled by the application of the χ^2 method of Dai, Hoang, and Tessaro in [DHT17]. We prove that mIFI is indistinguishable from a uniformly random αn-bit string with n-bit security. Up to our knowledge, this is the first fixed-length TPRF composition result to achieve arbitrary but fixed expansion with n-bit security.

Novel, fixed-length TPRF primitive. The mIFI approach allows us to *efficiently reuse* existing, well-analyzed and optimized components, such as TBCs, to instantiate our ButterKnife design (in Section 4). Our approach is reminiscent of the n-to-n-bit AES-based (T)PRF [DIS+18] method. We choose to base our design on the AES-based Deoxys TBC components due to its robustness in terms of security and its reliance on the AES-based internal structure. Deoxys-BC is used in both ZAE and the Deoxys [JNPS18, JNPS21] AE designs and the latter was selected as a finalist and first choice for the 'in-depth security' portfolio of the CAESAR competition.

ButterKnife uses a 256-bit tweakey and applies 7 Deoxys-BC-256 rounds until the forking point and another 8 rounds in each of the $\alpha = 8$ fully *parallelizable* branches (see Figure 1) to expand an n-bit input to an $8n$-bit output.

Cryptanalysis. ButterKnife benefits from the cryptanalysis of Deoxys-BC-256. However, due to the feed-forward and of the 8 output branches in ButterKnife, the security arguments of Deoxys-BC-256 do not directly apply to ButterKnife. Due to the structural similarity ButterKnife also benefits from the cryptanalysis of AES-PRF [MN17b]. We establish our design choices by providing a detailed

security analysis (in Section 5) of our proposal against well-known techniques namely, differential, linear, impossible differential and rectangle attacks.

Encryption, authentication and DAE algorithms. Towards secure DAE, we propose encryption and authentication schemes of independent interest and prove their n-bit security (for TPRFs with tweaks of size $t \geq n$). We introduce the CTR-style IV-based encryption scheme FEnc that uses $(n+\min(n,t))$-bit IVs and encrypts on average $m = \alpha n$ bits of plaintext for each call to the underlying α-blocks expanding TPRF. We also construct and prove secure the new PRF algorithm called SFMac that takes as input the data inputs (A, M) and the fixed key K to produce a tag of length $2n$ bits. The SFMac design uses a GHash-like construction and spends about one multiplication in $\mathsf{GF}(2^{2n})$ to process $2n$ bits of data and two calls to the underlying TPRF for the total data processing. The encryption scheme distinguishes itself by the optimized message processing due to the use of αn bit outputs from the TPRF (versus n-bit outputs for TBCs). Both schemes also support reducing the tag and IV length to any $\lambda \leq 2n$, which reduces the security level to $\min(\lambda, n + \min(n,t))/2$ bits.

We then combine both the latter authentication and encryption algorithms under the SIV composition [RS06] to achieve the n-bit secure DAE scheme called SAFE, that is inspired by the design of GCM-SIV. Since some platforms do not offer instructions to speed-up finite field multiplication, we also introduce the ZAFE DAE scheme, that combines the TPRF-based encryption from FEnc to the efficient TBC-based ZMAC MAC algorithm. Additionally, we choose all primitives so that they are parallelizable (and hence SAFE and ZAFE). Our proofs rely on the H coefficients technique and take advantage of the additional input space that comes from the tweak and of the larger output space to achieve full security using a number of primitive calls that is as small as possible.

Efficiency. We implement SAFE and compare it to state-of-the-art DAE schemes. Our detailed performance results are demonstrated in Table 5. Owing to the large TPRF output at reduced computational cost, our fully parallelizable encryption pass FEnc features significant improvements in throughput ($\approx$ 62-74%) with respect to the TBC-based encryption pass in other schemes such as ZAE. In the scope of n-bit secure DAEs, we observe our proposals SAFE and ZAFE to compare favorably to their TBC-based counterparts. Our SFMac implementation shows that GHash-style multiplication in $\mathsf{GF}(2^{2n})$ may, depending on the platform, contribute to no less effective SFMac processing than, e.g., ZMAC. Globally, SAFE and ZAFE maintain n-bit security, like ZAE, while moving much closer to AES-GCM-SIV in terms of performance (e.g., with 1.06 c/B on Skylake).

1.2 Organization

In Section 2, we recall the notions that are necessary for the rest of the paper. In Section 3, we introduce the masked-Iterate-Forf-Iterate paradigm and prove its soundness. In Sections 4 and 5, we define and analyze the ButterKnife expanding TPRF. Finally, in Section 6, we discuss the actual performance of our construc-

tions, and we illustrate the usefulness of expanding TPRF with our two modes of operation SAFE and ZAFE.

2 Preliminaries

2.1 General notation

For every positive integer n, we denote by $\{0,1\}^n$ the set of all n-bit binary strings, and by $(\{0,1\}^n)^+$ the set of all bit strings whose size is a non-zero multiple of n. The set of all bit strings will be denoted by $\{0,1\}^*$, and the empty string by ϵ. For any M in $\{0,1\}^*$, $|M|$ will be the bit length of the string M. Moreover, $M[1]||\dots||M[m] \xleftarrow{n} M$ means that we split M into m blocks of exactly n-bit strings, where $m = \lceil |M|/n \rceil$. If $M \notin (\{0,1\}^n)^+$, the one-zero padding is used on M beforehand. Finally, for any T in $\{0,1\}^*$, $[T]_n$ denotes the first n bits of T if $|T| \geq n$, or $T||0^{n-|T|}$ otherwise. Similarly, $M[1]||M[2] \xleftarrow{(n,t)} M$ means that we split the $(n+t)$-bit string $[M]_{n+t}$ into 2 blocks, where $M[1]$ is a n-bit string and $M[2]$ a t-bit string.

The set of all permutations of $\{0,1\}^n$ will be written $\mathrm{Perm}(n)$. Similarly, for every positive integer m, t such that $m \geq n$, the set of all tweakable permutations of $\{0,1\}^n$ with tweak space $\{0,1\}^t$ will be denoted by $\widetilde{\mathrm{Perm}}(t,n)$, and the set of all tweakable functions from $\{0,1\}^n$ to $\{0,1\}^m$ will be denoted by $\mathrm{Func}(t,n,m)$. For any keyed tweakable primitive $P : \mathcal{K} \times \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$ with key space $\mathcal{K}$, tweak space $\mathcal{T}$, domain $\mathcal{X}$ and range $\mathcal{Y}$, we will indifferently write $P(k,t,x)$, $P_k(t,x)$ or $P_k^t(x)$ for every tuple (k,t,x) in $\mathcal{K} \times \mathcal{T} \times \mathcal{X}$.

As usual, for any positive integers a, b such that $a \geq b$, we denote by $(a)_b$ the falling factorial $a(a-1)\cdots(a-b+1)$, with the convention that $(a)_0 = 1$.

Let $\mathrm{GF}(2^n)$ be the field of order 2^n. We identify n-bit strings and finite field elements of $\mathrm{GF}(2^n)$ by representing the string $a = a_{n-1}a_{n-2}...a_1a_0$ as the polynomial $a(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + ... + a_1x + a_0$ and vice versa. For any a, b in $\{0,1\}^n$,we define their sum $a \oplus b$ as the sum of the polynomials $a(x)+b(x)$. The product $a \otimes b$ or ab is defined with respect to the irreducible polynomial $f(x)$ used to represent $\mathrm{GF}(2^n)$ as $a(x) \cdot b(x) \mathrm{mod} f(x)$. Therefore, we can view $\{0,1\}^n$ as the finite field $\mathrm{GF}(2^n)$ with $\oplus$ as field addition and $\otimes$ as field multiplication. Sometimes, we also identify n-bit strings with integers in $\{0,\dots,2^n-1\}$. In that case, $\boxplus$ and $\boxminus$ denote the addition and subtraction modulo 2^n. Moreover, we define the $\oplus_t$ operation as follows: for any $x \in \{0,1\}^n$, and any $y \in \{0,1\}^t$, $x \oplus_t y = [x]_t \oplus y$. Note that one always has $|x \oplus_t y| = t$.

2.2 The H coefficients technique

In this section, we present the H coefficients technique [Pat08], which will be used to prove some of our results.

This technique is useful to upper bound the advantage of a (computationally unbounded and deterministic) distinguisher trying to distinguish between two worlds: a real world and an ideal one. Let D be such a distinguisher. In both

worlds, it has access to a tuple of oracles that have the same signature. Let us denote by O_{re} (resp. O_{id}) the tuple of real (resp. ideal) world oracles. Then, the advantage of D is defined as

$$\mathrm{Adv}(D) = |\Pr\left[D^{O_{\mathrm{re}}} = 1\right] - \Pr\left[D^{O_{\mathrm{id}}} = 1\right]|.$$

The interaction of D with its oracles will be summarized in the queries transcript τ of the attack. We are going to introduce two new random variables θ_{id} (resp. θ_{re}) which correspond to the transcript that is obtained by the interaction of D with the ideal (resp. real) world oracle. A transcript τ will be said *attainable* if and only if $\Pr[\theta_{\mathrm{id}} = \tau] > 0$, i.e. the probability of τ appearing in the real world is non-zero. The set of all attainable transcripts will be denoted by Θ. Then, one has the following classical lemma.

Lemma 1 ([Pat08,CS14]). *Let $\Theta_{\mathrm{bad}}, \Theta_{\mathrm{good}}$ be two sets such that $\Theta = \Theta_{\mathrm{bad}} \sqcup \Theta_{\mathrm{good}}$. If we assume that, for every transcript τ in Θ_{good}, one has*

$$\frac{\Pr[\theta_{\mathrm{re}} = \tau]}{\Pr[\theta_{\mathrm{id}} = \tau]} \geq 1 - \varepsilon,$$

then one has $\mathrm{Adv}(D) \leq \Pr[\theta_{\mathrm{id}} \in \Theta_{\mathrm{bad}}] + \varepsilon$.
As usual, the set Θ_{bad} (resp. Θ_{good}) will be referred to as the set of *bad* (resp. *good*) transcripts.

2.3 The χ^2 technique

In [DHT17], Dai, Hoang, and Tessaro introduced a new technique, dubbed the χ^2 method, to upper bound the statistical distance between the probability distributions of two sequences of random variables by computing the expectation of the χ^2 distances of the corresponding conditional distributions of the random variables. The core result of this technique is Lemma 2 given below. So far, it has been used to prove the security of various well-known constructions, such as the XORP constructions [DHT17, BN18],the encrypted Davies-Meyer construction [DHT17], or the Swap-Or-Not construction [DHT17].

Given a set Ω, let $\mathsf{X} := X^q := (X_1, \ldots, X_q)$ and $\mathsf{Y} := Y^q := (Y_1, \ldots, Y_q)$ be two random vectors distributed over Ω^q. For every $i \in [q]$, we write $\Pr_Z\left[z_i | z^{i-1}\right] := \Pr\left[Z_i = z_i | Z^{i-1} = z^{i-1}\right]$ for $Z \in \{\mathsf{X}, \mathsf{Y}\}$. For the special case when $i = 1$, we define $\Pr_Z\left[z_1 | z^0\right] := \Pr[Z_1 = z_1]$.

Definition 1. *Suppose that for every i and every y^i such that $\Pr\left[Y^i = y^i\right] > 0$, one also has $\Pr\left[X^i = y^i\right] > 0$. For every z^{i-1} in the support of Y^{i-1}, the χ^2-distance between these two conditional probability distributions is defined as*

$$\chi^2(z^{i-1}) := \sum_{x_i} \frac{\left(\Pr_{\mathsf{Y}}\left[x_i | z^{i-1}\right] - \Pr_{\mathsf{X}}\left[x_i | z^{i-1}\right]\right)^2}{\Pr_{\mathsf{X}}\left[x_i | z^{i-1}\right]},$$

where the sum is taken over all x_i such that $\Pr_{\mathsf{X}}\left[x_i | z^{i-1}\right] > 0$.

We also recall that the statistical distance between two distributions Pr_0 and Pr_1 whose supports are included in a set Ω is defined as follows:

$$\|\mathrm{Pr}_0 - \mathrm{Pr}_1\| = \frac{1}{2} \sum_{x \in \Omega} |\mathrm{Pr}_0(x) - \mathrm{Pr}_1(x)| .$$

Besides, the statistical distance also satisfies the following well-known property:

$$\|\mathrm{Pr}_0 - \mathrm{Pr}_1\| = \max_{X \subset \Omega} |\mathrm{Pr}_0(X) - \mathrm{Pr}_1(X)| .$$

We are now ready to state the core technical Lemma of the χ^2 method.

Lemma 2. *Assuming that the support of Y^i is included in the support of X^i for every $i = 1, \ldots, q$, one has*

$$\|\mathrm{Pr}_{\mathsf{X}} - \mathrm{Pr}_{\mathsf{Y}}\| \leq \left(\frac{1}{2} \sum_{i=1}^{q} \mathbb{E}\left[\chi^2(Y^{i-1})\right] \right)^{\frac{1}{2}} . \tag{1}$$

2.4 Security Notions

Tweakable pseudorandom functions/permutations. A TPRF is a keyed function $F : \mathcal{K} \times \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$. The tprf security of a TPRF is defined as follows.

Definition 2. *Let F be a TPRF. The advantage of an adversary A in breaking the* tprf*-security of F is defined as*

$$\mathrm{Adv}_F^{\mathrm{tprf}}(A) = \left| \Pr\left[A^{F_K} = 1\right] - \Pr\left[A^R = 1\right] \right| ,$$

where the probabilities are taken over the random choices of A and the uniformly random draw of K from $\mathcal{K}$ and R from the set of all functions from $\mathcal{T} \times \mathcal{X}$ to $\mathcal{Y}$. When $\mathcal{T} = \emptyset$, we recover the standard prf *security notion.*

It is easy to see that there is no difference between a secure TPRF with tweak space $\mathcal{T}$ and input space $\mathcal{X}$ and a secure PRF with input space $\mathcal{T} \times \mathcal{X}$. We adopt the tweakable formalism as the two inputs will have a different impact on performance in our instantiation, and thus it makes sense to distinguish them. When F is a TPRP, we define its tprp security in a similar way.

Definition 3. *Let F be a TPRP. The advantage of an adversary A in breaking the* tprp*-security of F is defined as*

$$\mathrm{Adv}_F^{\mathrm{tprp}}(A) = \left| \Pr\left[A^{F_K} = 1\right] - \Pr\left[A^P = 1\right] \right| ,$$

where the probabilities are taken over the random choices of A and the uniformly random draw of K from $\mathcal{K}$ and P from the set of all tweakable permutations from $\mathcal{T} \times \mathcal{X}$ to $\mathcal{X}$.

Deterministic AE. A deterministic authenticated encryption (DAE) scheme is a tuple $\mathsf{DAE} = (\mathcal{K}, \mathcal{AD}, \mathcal{M}, \mathcal{C}, \mathsf{DAE.Enc}, \mathsf{DAE.Dec})$ where $\mathcal{K}, \mathcal{AD}, \mathcal{M}, \mathcal{C}$ are non-empty sets and $\mathsf{DAE.Enc}, \mathsf{DAE.Dec}$ are deterministic algorithms such that:

- $\mathsf{DAE.Enc}$ takes as input a key K in $\mathcal{K}$ and a plaintext M in $\mathcal{M}$ with some associated data A in $\mathcal{AD}$, and returns a ciphertext C in $\mathcal{C}$;
- $\mathsf{DAE.Dec}$ takes as input a key K in $\mathcal{K}$, a ciphertext C in $\mathcal{C}$ and some associated data A in $\mathcal{AD}$, and returns either a plaintext M in $\mathcal{M}$, or the special symbol $\perp$ if the ciphertext is invalid;
- for any tuple (K, A, M) in $(\mathcal{K}, \mathcal{AD}, \mathcal{M})$, one has

$$\mathsf{DAE.Dec}_K(A, \mathsf{DAE.Enc}_K(A, M)) = M.$$

The dae-security of a DAE scheme is defined as follows.

Definition 4. *Let* DAE *be a DAE scheme. The advantage of an adversary* A *in breaking the* dae*-security of* DAE *is defined as*

$$\mathrm{Adv}^{\mathrm{dae}}_{\mathsf{DAE}}(A) = \left|\Pr\left[A^{\mathsf{DAE.Enc}_K, \mathsf{DAE.Dec}_k} = 1\right] - \Pr\left[A^{\$(\cdot,\cdot), \perp(\cdot,\cdot)} = 1\right]\right|,$$

where oracle $\perp$ *always returns* $\perp$*, and the probabilities are taken over the random choices of* A*, the uniformly random draw of* K *from* $\mathcal{K}$*, and the randomness of the oracle* \$ *which returns a uniformly random bit string of the same length as the corresponding output of* $\mathsf{DAE.Enc}$*.*

IV-based encryption scheme. An IV-based encryption scheme is a tuple $\mathsf{IVE} = (\mathcal{K}, \mathcal{IV}, \mathcal{M}, \mathcal{C}, \mathsf{IVE.Enc})$ where $\mathcal{K}, \mathcal{IV}, \mathcal{M}, \mathcal{C}$ are non-empty sets and $\mathsf{IVE.Enc}$ is a deterministic algorithm such that $\mathsf{IVE.Enc}$ takes as input a key $K \in \mathcal{K}$, an IV $I \in \mathcal{IV}$ and a plaintext $M \in \mathcal{M}$, and returns a ciphertext $C \in \mathcal{C}$. We denote by $\mathsf{IVE}^{\$}$ the associated randomized algorithm that starts by drawing a uniformly random IV from $\mathcal{IV}$.

The ive-security of an IV-based encryption scheme is defined as follows.

Definition 5. *Let* IVE *be an IV-based encryption scheme. The advantage of an adversary* A *in breaking the* ive*-security of* IVE *is defined as*

$$\mathrm{Adv}^{\mathrm{ive}}_{\mathsf{IVE}}(A) = \left|\Pr\left[A^{\mathsf{IVE}^{\$}.\mathsf{Enc}_K} = 1\right] - \Pr\left[A^{\$(\cdot,\cdot)} = 1\right]\right|,$$

where the first probability is taken over the uniformly random draw of K *from* $\mathcal{K}$ *and oracle* \$ *returns a uniformly random bit string of the same length as the corresponding output of* $\mathsf{IVE.Enc}$*.*

3 The masked Iterate-Fork-Iterate paradigm

In this section, we introduce a new design paradigm, dubbed masked Iterate-Fork-Iterate (mIFI), that can be used to build TPRFs using tweakable block

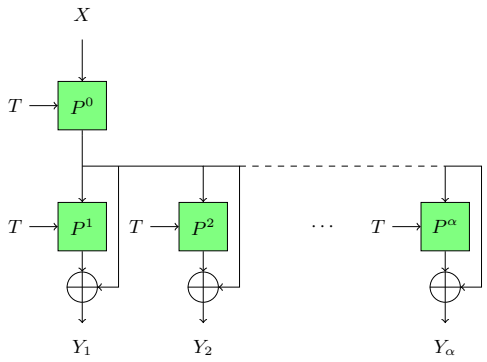

Fig. 1: Graphical representation of the mIFI paradigm, where P is a tweakable permutation with tweak space $[\alpha] \times \{0,1\}^t$.

cipher components. We will justify the soundness of our approach by proving the information-theoretical security of our generic construction, when instantiated with ideal primitives. As usual, this is not sufficient to justify the security of any instantiation of that paradigm, but it serves to rule out generic attacks.

The mIFI paradigm can be seen as a variant of the Iterate-Fork-Iterate (IFI) paradigm behind forkciphers: the main difference is that we XOR the forking state to all output blocks. A graphical depiction of this paradigm can be found in Figure 1. A property of forkciphers is that each of their branches behave as a tweakable block cipher: this inherently limits the TPRF-security of forkciphers to the birthday bound. As we will see, our simple change is sufficient to make the construction behave as an n-bit secure TPRF, where n denotes the size of the input block. It also allows us to keep the performance benefits of forkciphers, and to re-use components from existing primitives (tweakable block ciphers) to build our new algorithm. Of course, the feed-forward also precludes decryption and reconstruction queries that are inherent to forkciphers. We demonstrate in Section 6 AE schemes where no form of invertibitlity is required and hence a TPRF is a perfect fit both security- and performance-wise.

We prove the following result.

Theorem 1. *Let α, q and n be three strictly positive integers such that $(\alpha+1)q < 2^n$ and let $\mathcal{T}$ be a non-empty set. We denote by F_0 the* mIFI *paradigm where $P \xleftarrow{\$} \widetilde{\mathrm{Perm}}([\alpha] \times \mathcal{T}, n)$. Then, for any distinguisher D against the* tprf*-security of F_0 that issues at most q queries to its oracle, one has*

$$\mathrm{Adv}^{\mathrm{tprf}}_{F_0}(D) \leq \frac{\sqrt{2(\alpha+1)}q}{2^n}.$$

Proof Strategy. In this proof, we focus on information-theoretical adversaries: these are computationally unbounded, and can be considered deterministic without loss of generality. Let D be any distinguisher against the tprf-security of F_0 that issues at most q queries to its oracle. We will assume (also w.l.o.g.) that

D does not make any redundant queries, and that it always issues exactly q queries..

The first step of the proof is to notice that, in the real world, P can be lazily sampled. In general, as long as a random tweakable permutation is queried on a new pair (T, X), the answer can be chosen uniformly at random outside of the set of the previous outputs with the same tweak value T. When a new query (T, X) is issued to F_0, since D does not repeat queries, X has never been queried to $P^0(T, \cdot)$. Hence, $Z_0 := P^0(T, X)$ is chosen uniformly at random outside of the set of the Z_0 from the previous queries with tweak T. Then, the values $Z_i := P^\alpha(T, Y_0)$ can be chosen uniformly at random from the set of values that are different from the previous Z_i values with the same tweak. Hence, we can see that F_0 is indistinguishable from the PRF F_1 defined as follows:

$$\begin{aligned} F_1 : \mathcal{T} \times \{0,1\}^n &\longrightarrow \{0,1\}^{\alpha n} \\ (T, X) &\longmapsto \|_{i=1}^{\alpha} P^0(T, X) \oplus P^i(T, X) \end{aligned}$$

Hence,

$$\mathrm{Adv}^{\mathrm{tprf}}_{F_0}(D) = \mathrm{Adv}^{\mathrm{tprf}}_{F_1}(D).$$

This construction can be seen as a variant of the well-studied XORP construction, where independent permutations are used for each permutation call (and a tweak parameter has been added). We will follow in parts the idea for the proof of [BN18] and their convention for denoting an m-tuple $(x_1, \ldots, x_m)$ as x^m. This will make the size of each tuple involved in the proof easier to follow. First, we are going to reveal additional information to the adversary:

- in the real world, the intermediate values Z_0 will be revealed alongside each query;
- in the ideal world, a dummy value will be chosen uniformly at random in a set of authorized values (namely the set of all values that do not create any collision with a previous query).

The random variable that corresponds to the transcript of the interaction of D with the real (resp. ideal) world oracle will be denoted S (resp. R), and we give a formal definition of both in Algorithms 1 and 2[6]. Note that the conditions $D_i[T] = \{0,1\}^n$ and $D = \{0,1\}^n$ cannot occur since we assume $(\alpha + 1)q < 2^n$[7], we include them for the correctness of the algorithm. We let Ω^q denote the set of all transcripts s^q such that $\Pr_{\mathsf{S}}[s^q] > 0$[8]. Such a transcript can be parsed as a list of q tuples of the form $(t_i, x_i, y_{1,i} \| \cdots \| y_{\alpha,i}, z_{0,i})$, where:

[6] Recall that D is deterministic. Hence, all adversarial queries can be computed from the outputs of the oracle. We still make the queries explicit in the transcript in order to make the proof easier to follow.

[7] As a consequence, for every $y \in \{0,1\}^{\alpha n}$, there exists at least one value z_0 such that the probability of getting (y, z_0) as an output is non-zero, both in the real and the ideal world.

[8] This set depends on the adversary D.

– (t_i, x_i) corresponds to D's i-th query, that deterministically depends on the values of $y_{k,j}$ and $z_{0,j}$ for $j < i$ and $k = 1, \ldots, \alpha$;
– $(y_{1,i} \| \cdots \| y_{\alpha,i}, z_{0,i})$ corresponds to the output of the F_1 construction, which means that, for every $1 \leq i < j \leq q$, if $t_i = t_j$, then $z_{0,i} \neq z_{0,j}$ and $y_{k,i} \oplus z_{0,i} \neq y_{k,j} \oplus z_{0,j}$ for $k = 1, \ldots, \alpha$.

It is clear that, for such a transcript, one also has $\Pr_{\mathsf{R}}[s^q] > 0$ as it would have been possible for the ideal-world oracle to give the same outputs, albeit with a different probability. Conversely, any transcript r^q such that $\Pr_{\mathsf{R}}[r^q] > 0$ also satisfies $\Pr_{\mathsf{S}}[r^q] > 0$. Hence, both probability distributions have the same support Ω^q, and we can apply the χ^2 technique to S and R in order to lower bound $\|\Pr_{\mathsf{S}} - \Pr_{\mathsf{R}}\|$. Given that

$$\mathrm{Adv}^{\mathrm{tprf}}_{F_0}(D) = \mathrm{Adv}^{\mathrm{tprf}}_{F_1}(D) \leq \|\Pr_{\mathsf{S}} - \Pr_{\mathsf{R}}\|,$$

it will allow us to conclude the proof of Theorem 1.

Algorithm 1 S is the random variable corresponding to the outputs created by the interaction of D with the real-world oracle defined here.

variables
Tables of sets of previous Tweakable Permutation Outputs $(D_i[T])_{i \in \{0,\ldots,\alpha\}, T \in \mathcal{T}}$
end variables

function INITIALIZE
  for $i \in \{0, \ldots, \alpha\}$ **do**
    for $T \in \mathcal{T}$ **do**
      $D_i[T] \longleftarrow \emptyset$
    end for
  end for
end function
function QUERY(T,X)
  for $i \in \{0, \ldots, \alpha\}$ **do**
    if $D_i[T] = \{0,1\}^n$ **then**
      return 0^n
    end if
    $Z_i \xleftarrow{\$} \{0,1\}^n \setminus D_i[T]$
    $D_i[T] \longleftarrow D_i[T] \cup \{Z_i\}$
  end for
  for $i \in \{0, \ldots, \alpha\}$ **do**
    $Y_i \longleftarrow Z_0 \oplus Z_i$
  end for
  return $(T, X, Y_1 \| \ldots \| Y_\alpha, Z_0)$
end function

In Supplementary Material A, we prove the following Lemma.

Lemma 3.

$$\|\Pr_{\mathsf{S}} - \Pr_{\mathsf{R}}\| \leq \frac{\sqrt{2(\alpha+1)}q}{2^n}.$$

4 Specification of ButterKnife

In this section we propose the concrete instantiation ButterKnife, which mainly utilizes the round function and tweakey scheduling of Deoxys-BC.

Algorithm 2 R is the random variable corresponding to the outputs created by the interaction of D with the ideal-world oracle defined here.

variables
Tables of sets of previous Tweakable Permutation Outputs $(D_i[T])_{i\in\{0,\ldots,\alpha\},T\in\mathcal{T}}$
end variables

function INITIALIZE
  for $i \in \{0,\ldots,\alpha\}$ **do**
    for $T \in \mathcal{T}$ **do**
      $D_i[T] \leftarrow \emptyset$
    end for
  end for
end function
function QUERY(T,X)
  for $i \in \{1,\ldots,\alpha\}$ **do**
    $Y_i \xleftarrow{\$} \{0,1\}^n$
  end for
  $D \longleftarrow D_0[T]$
  for $i \in \{1,\ldots,\alpha\}$ **do**
    $D \longleftarrow D \cup (Y_i \oplus D_i[T])$
  end for
  if $D = \{0,1\}^n$ **then**
    return 0^n
  end if
  $Z_0 \xleftarrow{\$} \{0,1\}^n \setminus D$
  $D_0[T] \longleftarrow D_0[T] \cup \{Z_0\}$
  for $i \in \{1,\ldots,\alpha\}$ **do**
    $D_i[T] \longleftarrow D_i[T] \cup \{Z_0 \oplus Y_i\}$
  end for
  return $(T, X, Y_1\|\ldots\|Y_\alpha, Z_0)$
end function

The Deoxys-BC is based on AES round function and thus inherits some of its security properties. The Deoxys-BC-based AE [JNPS18] was a candidate and one of the winners of the Caesar competition. So, extensive security analyses were made on both Deoxys and Deoxys-BC (for instance [CHP+17, LSG+20, ZDJ19], see Table 6), which confirm its resistance to a number of cryptanalysis techniques.

In order to inherit from both the performance and security arguments of Deoxys-BC, and to serve our design necessities simultaneously, we modified as few elements as possible in ButterKnife. Most notably, the beginning of the series of operations that an input X goes through to give any Y_j is exactly the same as Deoxys-BC-256 except for the round constants.

Specification. ButterKnife uses a 256-bit tweakey and is based on Deoxys-BC-256, P^0 contains 7 rounds and P^j $(1 \leq j \leq \alpha = 8)$ contains 8 rounds (thus 15 rounds out of which 7 common rounds are iterated to obtain the corresponding Y_j from X).

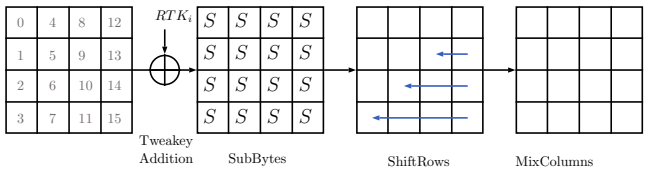

Fig. 2: One round transformation of the internal state of Deoxys-BC.

The 128-bit state of ButterKnife is represented as a 4×4 matrix of bytes. The tweakey is made of 256 bits, represented as two matrices in the same format. As represented in Figure 2, each round is made of the following operations:

- AddRoundTweakey (ART) adds the (128-bit) round tweakey to the internal state,
- SubBytes (SB) transforms each of the 16 bytes by applying the AES Sbox,
- ShiftRows (SR) rotates the i th row to the left by i positions, where $i \in \{0, 1, 2, 3\}$,
- MixColumns (MC) corresponds to the multiplication in $\mathbb{F}_{2^8}$ of each column by the circulant matrix $M = circ(02, 03, 01, 01)$.

A final tweakey addition is done before the final feed forward leading to the Y_j, $1 \leq j \leq 8$. As in Deoxys, the last MixColumns is not omitted. The tweak and the key are handled together by following the tweakey framework [JNP14] proposed by Jean *et al.*

At round i, the round tweakey RTK_i is obtained by xoring the current state of the two tweakey words with a round constant depending on the round i and on the branch index j_b ($j_b = 0$ before branching, and $j_b = j$ (≥ 1)after branching for the branch leading to Y_j):

$$RTK_i = TK_i^1 \oplus TK_i^2 \oplus \mathsf{Rconst}_{j_b,i}.$$

The set of round constants is computed as follows, where `RC[i]` is the AES round constant in the ith round:

$$\mathsf{Rconst}_{j_b,i} = \begin{pmatrix} 1 & \texttt{RC[i]} & j_b & 0 \\ 2 & \texttt{RC[i]} & j_b & 0 \\ 4 & \texttt{RC[i]} & j_b & 0 \\ 8 & \texttt{RC[i]} & j_b & 0 \end{pmatrix}$$

The choice of the number of rounds for ButterKnife and the other parameters are decided by the security analysis (given in Section 5).

5 Security analysis of ButterKnife

In this section we review the different attack techniques and show that the recommended number of rounds for our instance is resilient against them. We consider the single key and *related-tweakey* adversarial model for our security analysis, and aim for 128-bit security. We recommend to take a number of rounds before and after branching that is equal or really close and that every output Y_j is built from the same number of rounds.

Note that any one branch of ButterKnife, from X to Y_j ($1 \leq j \leq 8$), follows the FastPRF construction [MN17b] proposed by Mennink and Neves. Consequently, parts of the analysis made on its concrete instance AES-PRF in [DIS+18] apply to our case.

Note that the attacks on ForkAES [ARVV18] which exploits the reconstruction query functionality do not apply to ButterKnife. This is due to the facts that each branch of ButterKnife is not a permutation of the input and there is no reconstruction functionality i.e. to obtain a Y_i from a Y_j (for $i \neq j$ and $i, j \geq 1$), is impossible in ButterKnife. We stress that our analysis takes into account that different output branches share the first 7 rounds and that up to our analysis no weakness is induced by this.

5.1 Differential distinguishers

First scenario. If we focus on one branch, the cipher follows the same transformations as Deoxys-BC. The final feed forward and the different round constants have no impact on the number of active Sboxes of a differential characteristic, and thus the bounds provided in [CHP+17] apply (see Table 2).

Let us consider the differential characteristic corresponding to a single branch, w.l.o.g. the first branch producing the outputs Y_1, Y_1' under two distinct tweaks T, T' respectively. Suppose there is a tweakey differential characteristic with probability p observed for 15-round of Deoxys-BC-256 i.e. before adding the feed-forward internal state in ButterKnife, given by $(\boldsymbol{\Delta}_{in} \rightarrow \boldsymbol{\Delta}_1 \rightarrow \ldots \boldsymbol{\Delta} \rightarrow \ldots \widetilde{\boldsymbol{\nabla}}^1)$ where $\boldsymbol{\Delta}$ is the difference in the internal state at the round before branching. Then any branch j will have a tweakey differential characteristic $\boldsymbol{\Delta}_{in} \rightarrow \boldsymbol{\Delta}_1 \rightarrow \ldots \boldsymbol{\Delta} \rightarrow \ldots \widetilde{\boldsymbol{\nabla}}^j \oplus \boldsymbol{\Delta}$ with the same probability p.

Table 2: Lower bounds on the number of active S-boxes in the related-tweakey model for Deoxys-BC-256 (from the simple model in [CHP+17]).

rounds	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Deoxys-BC-256	0	0	1	5	9	12	16	19	23	26	29	32	35	38

Table 3: Bounds on the number of active Sboxes in the second scenario.

rounds after branching	1	2	3	4	5	6	7	8
active Sboxes	4	16	22	22	23	24	25	26

Since the S-box used in ButterKnife is differentially 4-uniform, a characteristic with 22 or more active Sboxes does not allow to distinguish the cipher, and thus the previous bounds indicate that this attack scenario does not threaten ButterKnife.

Second scenario. A second possibility would be to obtain information from the propagation of a difference between different branches, say the one leading to Y_i and the one leading to Y_j, computed from the same input X. The specific

structure of ButterKnife implies that the difference only comes from the difference in the definition of P^i and P^j (see the blue differences on the left in Figure 3).

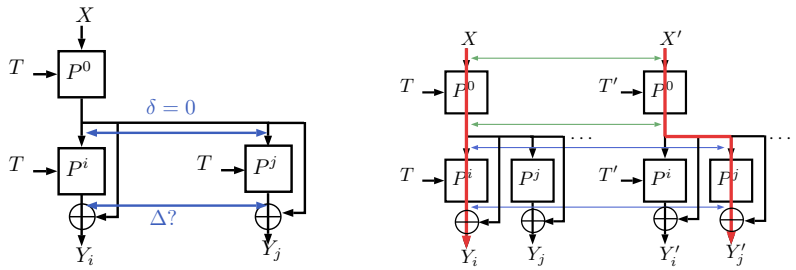

Fig. 3: Second and third scenario for a differential attack.

Right after branching the difference is naturally $\delta = 0$. The question is to determine if the variation between P^i and P^j is sufficient to avoid high probability differential characteristics.

From our definition of ButterKnife, the variation only comes from the constants added in the tweakey, so in each round the difference lies in the third column, and is equal to $(i \oplus j, i \oplus j, i \oplus j, i \oplus j)^T$.

A basic (byte-wise) MILP modelling of the problem returns the bounds in Table 3.

From this basic model we obtain that already after 3 rounds there are 22 active Sboxes, and then that a single characteristic would not allow to distinguish. As we discuss in Appendix E.1, these rather slowly increasing bounds are not tight.

Third scenario. We can also consider two inputs X and X' (processed under two tweakeys T and T'), and look at the difference propagation from the inputs to two different branches of each process. For instance, a difference starting from $X \oplus X'$ to $Y_i \oplus Y'_j$. (see on the right of Figure 3).

The first part of a characteristic in this setting follows the same constraints as a characteristic over Deoxys-BC, and thus the same bounds as the one given in Table 2 apply on this portion. The framework is more complicated for the lower part, for which we may have a difference in the internal state, in the tweak, and coming from the round constants.

We again made a byte-wise MILP model to get an idea of the number of active Sboxes. The results are reported in Table 4. Note that we did not force a difference in the tweakey nor in the input so that the minimum returned here might correspond to characteristics following the previous scenario with no difference in X nor in T.

Table 4: Bounds on the number of active Sboxes in the third scenario.

rounds before + after branching	2+2	3+3	4+4	5+5	6+6	7+7	7+8
active Sboxes	2	7	17	22	24	25	26

5.2 Linear trails

Any 4 consecutive rounds of ButterKnife have the same property as Deoxys and as the AES, that is, activate at least 25 Sboxes. Moreover, the impact of a tweak on the resistance to linear cryptanalysis was studied in detail in [KLW17], with the important conclusion that "tweaking a block cipher with a linear tweak schedule does not introduce any new linear trails".

5.3 Meet-in-the-Middle attacks

A MitM attack was presented in [DIS$^+$18] against the AES-PRF [MN17b]. This analysis shows that 7 rounds (with s rounds before and $7-s$ rounds after the feed-forward) of AES-PRF can be attacked with time and (chosen plaintext) data complexity 2^{107}. A single branch in ButterKnife resembles the FastPRF (or AES-PRF) construction. The attack in [DIS$^+$18] can also be applied against a single branch of ButterKnife. However, this is only able to break 7 out of 15 rounds. The MiTM attack [LJ19] on Deoxys-BC-256 (with 128-bit key) breaks 8 rounds of the cipher. This attack is based on the same distinguishing technique (from Demirci and Selçuk [DS08]) that is also used in [DIS$^+$18]. For ButterKnife (with a 128-bit key), it is possible to break a maximum of 8 rounds.

5.4 Impossible differentials and zero-correlation

First case (Single Branch). As for the differential, we start with the scenario that studies a single branch (X to one of the Y_i), which corresponds to the FastPRF construction. The impossible differential technique used against AES-PRF [DIS$^+$18] can be applied to any branch of ButterKnife, for a version with P^j $(1 \leq j \leq 8)$ of any number of rounds. This is depicted in Figure 4.

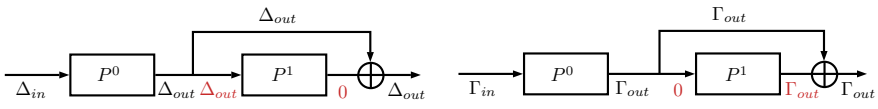

Fig. 4: Impossible differential and Zero-correlation attacks (from [DIS$^+$18]).

In the single tweakey scenario, P^j is a permutation and as such cannot cancel a difference: for any non-zero input difference Δ_{out} the output difference cannot

be 0. Thus, if the output difference of P^0 is Δ_{out} it is impossible to observe Δ_{out} after the feed forward. This is an impossible differential distinguisher, and it can be converted into an attack by positioning the key-recovery rounds in P^0.

An impossible differential attack [DIS+18] against AES-PRF$_{2,8}$ i.e., 2 rounds before and 8 rounds after the feed forward step, can be mounted. The full ButterKnife, i.e. with 7 and 8 rounds before and after feed forward respectively, seems secure against this type of attack as we expect that no more than 3 key recovery rounds can be added.

A similar technique can be used to create a zero-correlation linear distinguisher (depicted on the right in Figure 4) that leads to a ZC attack against AES-PRF$_{2,8}$. We expect that the same number of rounds of ButterKnife can be attacked in this fashion.

We note that like different configurations of AES-PRF in [DIS+18], it is possible to attack different round-reduced versions of ButterKnife. For example, corresponding to AES-PRF$_{6,4}$, 6 and 4 rounds before and after feed-forward in ButterKnife, corresponding to AES-PRF$_{7,3}$, 7 and 3 rounds before and after feed-forward are possible to attack using zero-correlation attack. However, the full ButterKnife is secure against such attacks.

Second scenario. The second possible scenario studies two different outputs Y_i and Y_j, and thus looks for impossible differences between these two branches. By searching for a trail in a miss in the middle way, we obtain the 3-round impossible differential trail given in Supplementary Material, in Figure 10.

5.5 Amplified boomerang distinguishers

Given that decryption queries are impossible, we focus on the chosen plaintexts variant of the boomerang attack, that is, on amplified boomerangs [KKS01].

An amplified distinguisher works by splitting the cipher E into two parts, $E = E_1 \circ E_0$ and by using a differential for each: a first differential of probability p over E_0, and a second one over E_1 of probability q. The distinguisher uses quartets of messages (X^i, $0 \leq i \leq 3$) with two fixed input differences ($X^0 \oplus X^1$ and $X^2 \oplus X^3$), and count the number of times a specific difference appears both in $E(X^0) \oplus E(X^2)$ and in $E(X^1) \oplus E(X^3)$. A first estimate gives the probability of a distinguisher based on the previous differentials to be $p^2q^2 2^{-n}$, which has to be compared to 2^{-2n} for the ideal case.

Once again, several scenarios are possible[9]. The first one (on the left in Figure 5) corresponds to constructing quartets with 4 independent branches built from 4 different inputs ((X^i, Y^i_j) for a fixed j and $0 \leq i \leq 3$). In the best case, it returns the same results as for Deoxys-BC (at the time of writing the best published distinguisher reaches 9 rounds in the related-tweak scenario [ZDJ19]). A second possibility would be to consider the setting depicted on the right of Figure 5 for which a quartet is obtained from 4 different inputs and by looking at two different branches (number $i \geq 1$ and $j \geq 1$ in the figure). By using the

[9] We first discuss the case where the feed forward is omitted

bounds returned by the MILP models for the various scenarios, the higher number of rounds that can be attacked in this setting is 10: the upper trail would cover a part E_0 made of 3 rounds of P^0, while the lower trail would cover the next 3 rounds of P^0 and 4 rounds of the bottom part. Namely, the first differential characteristic would be in the first differential scenario (3 rounds, 1 active Sbox), and the second differential characteristic would be in the third differential scenario and cover 3 rounds before and 4 rounds after branching, with 9 active Sboxes. Assuming our bounds are tight[10] and that each Sbox transition can be done with probability 2^{-6}, it would result in a distinguisher of probability $p^2q^2 2^{-n} = 2^{-12}2^{-108}2^{-128} = 2^{-248}$.

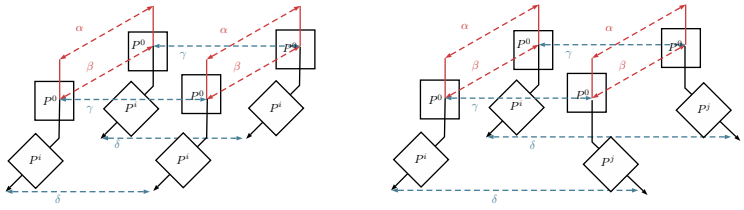

Fig. 5: Setting 1 and 2 for a boomerang attack.

Another possible setting would be to only consider two different inputs and two of their branches, as depicted in Figure 6. On the left, we consider the first differential to be starting from the 0 difference, right after branching, while the second differential characteristic is put between the two branches of equal index. A maximum of 12 rounds can be covered in this fashion: 1 round (with 4 active Sboxes) after branching, followed by 4 rounds (with 5 active Sboxes) and since any number of rounds can be added on top (before branching), all the 7 of them can be included. Such a distinguisher would have a probability around $2^{-128-108}$. However, a problem with this scenario is the difficulty to detect the boomerang once the feed forward is taken into account. To work around this, we consider another setting (depicted on the right in Figure 6) where the characteristic over E_0 is nonexistant: we consider that a certain difference γ appears between the two branching point of two different states and use a (or two different) characteristic(s) from the branching point to the two outputs of equal index, leading to the difference Δ (or Δ_1 and Δ_2) before the feed forward, and equal to $\Delta \oplus \delta$ (or $\Delta_1 \oplus \delta$ and $\Delta_2 \oplus \delta$) after it. Again, the 7 rounds before branching can be included, and 5 rounds can be considered after branching with 10 active Sboxes, thus reaching a probability close to $2^{-128-120}$.

Given that ButterKnife has 8 rounds after branching, we believe that only reduced round versions could be attacked with this technique.

[10] Arguments similar to the ones shown in supplementary material E.1 apply here and show that we are not tight.

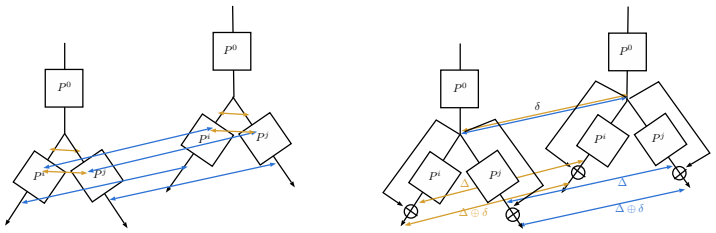

Fig. 6: Setting 3 and 4 for a boomerang attack.

5.6 Other Aspects

In some parts of the previous discussion, we mostly focused on distinguishers without taking into account the final feed-forward. However, this ending operation makes difficult to add a key recovery part at the bottom. The key recovery requires to invert rounds in value to access the end of the distinguisher, which is made difficult by the unknown value of the middle internal state.

When given the full code book a zero-sum distinguisher can be constructed against one or more branch in the same way as described in [MN17b, Appendix A.1] with distinguishing advantage $1 - \frac{1}{2^{128}}$. In an integral attack, adding a round tweakey does not have any impact. This allows an adversary to apply the integral attacks on reduced-round AES to round-reduced ButterKnife also.

As already mentioned, one could try to apply the techniques used for the cryptanalysis of ForkCiphers [BBJ+19], but with the difference that the reconstruction setting is not available given that one cannot make decryption queries. A reflection differential distinguishers might exist between two branches, but accessing it and building the required pair is troublesome.

6 DAE for TPRFs: Security and Performance

6.1 Presentation

This section focuses on DAE applications of *expanding* TPRFs as an important cryptographic block in security. A TPRF with input space and tweak space $\{0,1\}^n$, and output space $\{0,1\}^{\alpha n}$ with $\alpha \geq 2$ is of interest when its evaluation is faster than the computation of α outputs of an equivalently secure TBC or non-expanding TPRF. In particular, this means that it excels in counter-mode scenarios, where the goal is to output a keystream as fast as possible. Conversely, it does not appear to be a good fit for data absorption. Hence, in order to build a DAE as the union of fast encryption and authentication schemes, our proposed authentication mechanisms will be TPRF independent. To this end, we propose 2 DAE constructions to efficiently tackle this goal:

– the SAFE DAE mode of operation that combines fast TPRF-based encryption in counter-mode with a polynomial-based hash function;

– the ZAFE DAE mode of operation that combines fast TPRF-based encryption in counter-mode with a MAC based on a tweakable block cipher.

While the encryption portion in both modes stays the same, we propose two distinct authentication methods to enable adequate support for a larger spectrum of applications and platforms.

Let us fix a TPRF $F : \mathcal{K} \times \{0,1\}^{t+1} \times \{0,1\}^n \to \{0,1\}^m$ with key space $\mathcal{K}$, tweak space $\{0,1\}^{t+1}$, domain $\{0,1\}^n$ and range $\{0,1\}^m$ for the remainder of this section, which is dedicated to the presentation and analysis of both modes of operation. As we will see, both SAFE and ZAFE offer the same security level of $(n + \min(n,t))/2$ bits, assuming that E offers the same tweak size as F. We then implement both schemes with ButterKnife and compare their performance with Deoxys-based modes of operations.

6.2 The SAFE mode of operation

Statement of the result and discussion. Let us also fix an integer λ such that $\lambda \leq 2n$. The SAFE mode of operation can be divided into two primitives:

1. a PRF algorithm $\mathsf{SFMac}[F]$ that takes as input two messages (message and associated data), and processes on average $2n$ bits of input for every multiplication in $\mathsf{GF}(2^{2n})$. $\mathsf{SFMac}[F]$ uses a single call to F during the finalization, and outputs a λ-bit tag;
2. an IV-based encryption scheme FEnc that uses $\min(\lambda, n + t)$-bit IVs and encrypts on average m bits of plaintext for each call to F, using a constant tweak value.

We combine both primitives using the SIV generic structure [RS06] in order to construct our DAE scheme. It is also worth noting that one of our motivation for the choice of underlying primitives was the fact that the calls to F in the encryption pass can be computed in parallel, and SFMac algorithm relies on a GHash-like construction, which can also be computed in parallel.

The design of SAFE is inspired by the design of $\mathsf{GCM-SIV}$. Most improvements were possible thanks to the fact that the output of a TPRF can be larger than its input. In more details, the encryption of an (A, M) pair is done as follows. A $2n$-bit hashing key L is first derived from the key using a call to F with the all-zero tweak and message as its inputs. Then, the associated data A and message M are parsed into $2n$-bit blocks. A $2n$-bit hash Y of (A, M) is computed, the tag is simply

$$T = [F_K^{0||Y[2]}(Y_1)]_\lambda,$$

where $Y_1||Y_2 \xleftarrow{n,t} Y$. This tag is then used as IV in $\mathsf{FEnc}[F]$, which behaves similarly to the counter mode of operation for block ciphers but when instantiated with a TPRF.

The formal specification of SAFE is presented in Algorithm 3. Moreover, a graphical representation of the encryption algorithm can be found in Figure 7

in supplementary material. We prove the security of $\mathsf{SAFE}[F]$ via the following theorem.

Theorem 2. *Let $n, m, t, \sigma, q, \lambda$ be positive integers such that $m \geq 2n \geq 1$, $q, \sigma < 2^n$ and $\lambda \leq 2n$. Let F be a TPRF with tweak space $\{0,1\}^{t+1}$, domain $\{0,1\}^n$ and range $\{0,1\}^m$. Let also D be an adversary against the* dae*-security of $\mathsf{SAFE}[F]$ that runs in time at most* `time`*, and makes at most q queries for a total of at most σ n-bit blocks. Then there exists an adversary B against the* prf*-security of F that runs in time at most* `time`$+O(\sigma)$ *and makes at most $\sigma + q + 1$ chosen plaintext queries such that*

$$\mathrm{Adv}^{\mathrm{dae}}_{\mathsf{SAFE}[F]}(A) \leq \mathrm{Adv}^{\mathrm{tprf}}_{F}(B) + \frac{q}{2^{\lambda}} + \frac{2(q-1)\sigma}{2^{m'}} + \frac{2q\sigma + 2q^2 + \sigma + 4q}{2^{n+\min(n,t)}},$$

where $m' = \min(\lambda, n + t)$.

Proof. Due to space constraints, we defer the proof of Theorem 2 to Supplementary Material D.2, but we briefly sketch its main ingredients here. First, we start by applying a hybrid argument to replace F by its ideal counterpart. We then use a composition result to split the study of the dae-security of SAFE in two parts: the prf-security of its authentication pass, and the ive-security of its encryption pass. Both are then studied independently with the H coefficients technique, and rely on the fact that inputs to F only collide with negligible probabilities.

Remark 1. The parameter λ allows for some flexibility when choosing the tag length. As far as security is concerned, the bound from Theorem 2 tells us that $\lambda \geq n + \min(n, t)$ is a reasonable choice. When one has $t + 1 = n$ (i.e. the underlying TPRF has the same tweak length and block length, one bit of tweak being reserved for domain separation by SAFE), $\lambda = 2n$ might be preferable if n is a multiple of 8 since it will prevent having an unused bit in one byte of the tag.

6.3 The ZAFE mode of operation

The ZAFE mode of operation relies on a TBC $E : \mathcal{K} \times \{0,1\}^{t'} \times \{0,\ldots,9\} \times \{0,1\}^n \to \{0,1\}^n$ with key space $\mathcal{K}'$, tweak space $\{0,1\}^{t'} \times \{0,\ldots,9\}$, and block space $\{0,1\}^n$. Like in the case of SAFE, it can be seen as the composition through the SIV construction of two primitives:

- the PRF algorithm ZMAC that processes on average $n + t'$ bits of data per TBC call;
- the IV-based encryption scheme FEnc.

Note that, in this scenario, both primitives rely on different keys. This distinction was not necessary in the case of ZAE, as both components relied on the same TBC, and could be replaced by two independent tweakable permutations thanks to domain separation. A possible solution in our case would be to derive both keys from the same master key, by using calls to F in order to generate both subkeys. For the sake of simplicity, we present the 2-key version in Algorithm 4.

One has the following result.

Algorithm 3 The $\mathsf{SAFE}[F]$ Authenticated Encryption scheme, where $F_K \in \mathrm{Func}(t+1,n,m)$ for every key K. Here $\otimes$ denotes the multiplication in the field $\mathsf{GF}(2^{2n})$.

function $\mathsf{SFMac}(K,A,M)$
  $T \longleftarrow 0^{2n}$
  $L \longleftarrow \left[F_K^{0^{t+1}}(0^n)\right]_{2n}$
  $X \longleftarrow \mathrm{Pad}_{10}(A)||\mathrm{Pad}_{10}(M)$
  $X \longleftarrow X||\langle|A|\rangle_n||\langle|M|\rangle_n$
  $X[1]||\cdots||X[x] \xleftarrow{2n} X$
  for $i \longleftarrow 1,x$ **do**
    $T \longleftarrow (T \oplus X[i]) \otimes L$
  end for
  $U||V \xleftarrow{(n,t)} [T]_{n+t}$
  $T \longleftarrow F_K^{0||V}(U)$
  return $[T]_\lambda$
end function

function $\mathsf{SAFE.Enc}(K,A,M)$
  $T \longleftarrow \mathsf{SFMac}(K,A,M)$
  $\mathrm{IV} \longleftarrow [T]_{\min(\lambda,n+t)}$
  $C \longleftarrow \mathsf{FEnc}(K,\mathrm{IV},M)$
  return (C,T)
end function

function $\mathsf{FEnc}(K,I,M)$
  $U||V \xleftarrow{(n,t)} [I]_{n+t}$
  $C[1]||\cdots||C[c] \xleftarrow{m} M$
  for $i \longleftarrow 1,c$ **do**
    $C[i] \longleftarrow C[i] \oplus [F_K^{1||V}(U \boxplus i-1)]_{|C[i]|}$
  end for
  return $C[1]||\cdots||C[c]$
end function

function $\mathsf{SAFE.Dec}(K,A,C,T)$
  $\mathrm{IV} \longleftarrow [T]_{\min(\lambda,n+t)}$
  $M \longleftarrow \mathsf{FEnc}(K,\mathrm{IV},C)$
  $T' \longleftarrow \mathsf{SFMac}(K,A,M)$
  if $T = T'$ **then**
    return M
  else
    return $\perp$
  end if
end function

Algorithm 4 The $\mathsf{ZAFE}[F,E]$ Authenticated Encryption scheme, where $F_K \in \mathrm{Func}(t+1,n,m)$ for every key K, and $E_{K'} \in \mathrm{Perm}(\{0,1\}^{t'} \times \{0,\ldots,9\},n)$. The full ZMAC pseudocode is given in Supplementary Material C.

function $\mathsf{ZFMac}(K,A,M)$
  $X \longleftarrow \mathrm{Pad}_{10}(A)||\mathrm{Pad}_{10}(M)$
  $X \longleftarrow X||\langle|A|\rangle_n||\langle|M|\rangle_n$
  $T \longleftarrow \mathsf{ZMAC}(K,X)$
  return $[T]_\lambda$
end function

function $\mathsf{ZAFE.Enc}(K,K',A,M)$
  $T \longleftarrow \mathsf{ZFMac}(K',A,M)$
  $\mathrm{IV} \longleftarrow [T]_{\min(\lambda,n+t)}$
  $C \longleftarrow \mathsf{FEnc}(K,\mathrm{IV},M)$
  return (C,T)
end function

function $\mathsf{ZAFE.Dec}(K,K',A,C,T)$
  $\mathrm{IV} \longleftarrow [T]_{\min(\lambda,n+t)}$
  $M \longleftarrow \mathsf{FEnc}(K,\mathrm{IV},C)$
  $T' \longleftarrow \mathsf{ZFMac}(K',A,M)$
  if $T = T'$ **then**
    return M
  else
    return $\perp$
  end if
end function

Theorem 3. *Let $n, m, t, \sigma, q, \lambda$ be positive integers such that $m \geq 2n \geq 1$, $q, \sigma < 2^n$ and $\lambda \leq 2n$. Let F be a TPRF with tweak space $\{0,1\}^{t+1}$, domain $\{0,1\}^n$ and range $\{0,1\}^m$, and let $E : \mathcal{K} \times \{0,1\}^{t'} \times \{0,\ldots,9\} \times \{0,1\}^n \to \{0,1\}^n$ be a TBC with key space $\mathcal{K}'$, tweak space $\{0,1\}^{t+1} \times \{0,\ldots,9\}$, and block space $\{0,1\}^n$. Let also D be an adversary against the* dae*-security of* $\mathsf{ZAFE}[F,E]$ *that runs in time at most* `time`*, and makes at most q queries for a total of at most σ n-bit blocks. Then there exist an adversary B against the* tprf*-security of F an adversary C against the* tprp*-security of E that both run in time at most* `time` $+ O(\sigma)$ *and make at most $\sigma + 4q + 2$ chosen plaintext queries, such that*

$$\begin{aligned}&\mathrm{Adv}^{\mathrm{dae}}_{\mathsf{ZAFE}[F,E]}(A)\\ &\quad\leq \mathrm{Adv}^{\mathrm{tprf}}_{F}(B) + \mathrm{Adv}^{\mathrm{tprp}}_{E}(C) + \frac{q}{2^{\lambda}} + \frac{2(q-1)\sigma}{2^{m'}} + \frac{2.5\sigma^2}{2^{n+\min(n,t')}} + 4\left(\frac{q}{2^n}\right)^{3/2},\end{aligned}$$

where $m' = \min(\lambda, n+t)$.

Proof. The proof of Theorem 3 is very similar to the proof of Theorem 2, and we only highlight the differences here. We first start by applying a hybrid argument to replace both F and E by their ideal counterparts. Then, the variant of [RS06, Theorem 1] allows us to separate the study of the authentication pass and the encryption pass. Since the latter simply corresponds to FEnc, we can directly reuse Theorem 6 to upper-bound its ive-advantage. Finally, we upper-bound the prf-security of ZMAC with [IMPS17, Theorem 1]. Note that the fact that we truncate the output of ZMAC to λ bits does not reduce its security, since a prf-adversary against the truncated version can always be turned into a prf-adversary against the full version, with the same advantage, and a small time complexity overhead.

6.4 Implementation aspects

To evaluate the performance of SAFE and ZAFE, we implement the two principal novel components proposed in this work. First, we describe the implementation of our main result FEnc, the encryption pass based on ButterKnife. Second, we cover implementation aspects of SFMac as, prior to this work, it was uncertain whether an implementation in a larger field would maintain competitive performance [IMPS17].

Implementation of FEnc. Many contemporary devices support the AES round function in hardware (e.g., `AES-NI`). Similar to other parallel AES-based modes, SAFE and ZAFE can harness the full power of the `AES-NI` pipeline for sufficiently long messages [Gue09], or with task-level parallelism [BLT15]. In accordance with other implementations, we consider eight simultaneous AES-based primitives (*i.e.*, AES, Deoxys and ButterKnife). Within ButterKnife, all branches are computed independently, and round tweakeys can be precomputed. Furthermore, as the counter is given as an input block to ButterKnife, as opposed to the tweak, the tweakey-schedule only needs to be evaluated once for all calls to the primitive.

Implementation of SFMac. We generally rely on the same implementation techniques that were used for the GCM mode of operation [GK14]. Namely, we use the carry-less multiplication instructions (`PCLMULQDQ`) that are supported on many recent processors, as well as the fast reduction algorithm from [GK14] in order to implement the multiplication in the finite field GF(2^{256}). In order to keep the reduction algorithm efficient, we used the polynomial $x^{256}+x^{10}+x^5+x^2+1$. Moreover, we also aggregate the reduction step by computing the first λ powers of the hashing key, and by only reducing once every λ multiplications. Of course, the optimal value of λ depends on the message length and the microarchitecture. Based on experiments, we find $\lambda = 32$ to be an adequate aggregation level.

6.5 Evaluation

Table 5: Encryption performance of DAE modes in cycles per byte (c/B) for long messages (64KiB), decoupled in authentication and encryption passes. The ZAE instances use Deoxys-256 and/or Deoxys-384.

		Skylake (3.2 GHz)			Cascade Lake SP (2.7 GHz)		
Mode	**Security**	**Auth.**	**Enc.**	**Total**	**Auth.**	**Enc.**	**Total**
AES-GCM-SIV [GLL17]	64	0.27	0.56	0.83	0.21	0.42	0.63
SCT [PS16]	64	0.87†	0.87†	1.74†	0.60†	0.60†	1.20†
ZAE (256) [IMPS17]	128	0.61†	0.87†	1.48†	0.42†	0.60†	1.02†
ZAE (384) [IMPS17]	128	0.59†	0.99†	1.58†	0.41†	0.68†	1.09†
ZAE (256-384) [IMPS17]	128	0.59†	0.87†	1.46†	0.41†	0.60†	1.01†
SAFE	**128**	**0.56**	**0.50**	**1.06**	**0.42**	**0.37**	**0.79**
ZAFE	**128**	**0.60**†	**0.50**	**1.10**†	**0.42**†	**0.37**	**0.79**†

(† *are estimates based on measured counter-in-tweak performance*, cf. [IMPS17])

Comparison. Our own SAFE implementations are written in C, using compiler intrinsics for `AES-NI` and `PCLMULQDQ` instructions. For the encryption pass with ButterKnife, we adapt the Deoxys implementations from [JNPS16]. For AES-GCM-SIV, we use assembly-optimized implementations[11] from [GLL17], noting that assembly implementations may further improve the throughput of SAFE/ZAFE as well.

To our knowledge, there are no publicly available implementations of ZAE and SCT. However, as explored by Iwata et al. [IMPS17], the performance of

[11] Taken from `https://github.com/Shay-Gueron/AES-GCM-SIV` (September 2021)

their TBC-based building blocks is well-understood. We adopt their estimation methodology to estimate the performance of ZAFE, and compare our modes with other DAE schemes (ZAE and SCT). We confirm their long-message estimates for Deoxys counter-in-tweak on `Skylake` (*i.e.*, 0.87 c/B for Deoxys-256 and 0.99 c/B for Deoxys-384). For `Cascade Lake SP`, this yields estimates of 0.60 c/B for Deoxys-256 and 0.68 c/B for Deoxys-384. Like Iwata et al., for random tweak inputs, we apply a penalty of 1.4 for Deoxys-256 and 1.8 for Deoxys-384.

We use the `rdstc(p)` instructions on `x86` to measure performance across 50 batches, each averaging 500 measurements, of which the fastest batch is retained. To reflect practical scenarios and match prior work [GLL17], the measurement iterations are preceded by iterations that warm up the instruction and data caches.

In addition to `Skylake` (i5-6500, legacy, 2015), we also report evaluate performance of SAFE/ZAFE on the server-grade `Cascade Lake SP` (Xeon Platinum 8280, server, 2019). For reproducibility, we additionally make our implementations and measurements available in supplementary material.

Results. Table 5 gives encryption performance of SAFE/ZAFE w.r.t. comparable schemes. We decouple authentication and encryption passes, allowing them to be compared separately, and consider long messages (64 KiB in our experiments).

The encryption pass in SAFE/ZAFE benefits from two main design choices. First, ButterKnife itself has excellent performance due to a more efficient use of AES rounds (*i.e.*, on average 8.875 AES rounds per output block, compared to 14 or 16 for Deoxys). Second, the static tweakey schedule throughout the entire encryption pass compares favorably to a counter in the tweak. As a result, we observe an estimated throughput increase of $\approx 62-74\%$ w.r.t. counter-in-tweak with Deoxys-256 (SCT and ZAE) and $\approx 84-96\%$ w.r.t. Deoxys-384 (ZAE). ButterKnife is even competitive to AES in counter mode (cf. AES-GCM-SIV).

For the authentication pass (*i.e.*, MAC), Table 5 indicates that our SFMac implementation in $GF(2^{256})$ is approximately twice as slow as in $GF(2^{128})$, e.g., as used in AES-GCM-SIV. Our findings suggest that it is at least competitive to ZMAC [IMPS17], provided that the platform supports efficient field multiplications (e.g., `PCLMULQDQ` on x86). On platforms without such hardware support, we expect ZMAC to outperform SFMac (and hence, ZAFE to outperform SAFE).

Globally, SAFE/ZAFE outperform current state-of-the-art n-bit secure DAE schemes with an estimated 38% on `Skylake` and 28% on `Cascade Lake SP`.

7 Conclusion

In this work we presented the first n-to-αn TPRF mIFI design paradigm for $\alpha \geq 2$ with generic n-bit security. We presented a concrete instance of such TPRF, called ButterKnife and supported its security with extensive cryptanalysis. We then designed two provably secure DAE schemes: SAFE and ZAFE, which combine our fast TPRF-based encryption pass with two efficient authentication algorithms with $2n$-bit outputs. Both schemes come with approximately n-bit

security and give important efficiency improvements over ZAE when instantiated with ButterKnife.

We believe TPRFs can find further applications in scenarios where data expansion is used. We leave this research for future work.

References

ABPV21. Elena Andreeva, Amit Singh Bhati, Bart Preneel, and Damian Vizár. 1, 2, 3, fork: Counter mode variants based on a generalized forkcipher. *IACR Trans. Symmetric Cryptol.*, 2021(3):1–35, 2021.

AES01. Advanced Encryption Standard (AES). National Institute of Standards and Technology (NIST), FIPS PUB 197, U.S. Department of Commerce, November 2001.

ALP+19. Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. Forkcipher: A new primitive for authenticated encryption of very short messages. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part II*, volume 11922 of *LNCS*, pages 153–182. Springer, Heidelberg, December 2019.

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`.

BBJ+19. Subhadeep Banik, Jannis Bossert, Amit Jana, Eik List, Stefan Lucks, Willi Meier, Mostafizar Rahman, Dhiman Saha, and Yu Sasaki. Cryptanalysis of ForkAES. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 43–63. Springer, Heidelberg, June 2019.

BBLT18. Subhadeep Banik, Andrey Bogdanov, Atul Luykx, and Elmar Tischhauser. SUNDAE: Small universal deterministic authenticated encryption for the internet of things. *IACR Trans. Symm. Cryptol.*, 2018(3):1–35, 2018.

BJK+16. Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY family of block ciphers and its low-latency variant MANTIS. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 123–153. Springer, Heidelberg, August 2016.

BKR98. Mihir Bellare, Ted Krovetz, and Phillip Rogaway. Luby-rackoff backwards: Increasing security by making block ciphers non-invertible. In Kaisa Nyberg, editor, *EUROCRYPT '98, Proceeding*, volume 1403 of *LNCS*, pages 266–280. Springer, 1998.

BLT15. Andrey Bogdanov, Martin M. Lauridsen, and Elmar Tischhauser. Comb to pipeline: Fast software encryption revisited. In Gregor Leander, editor, *Fast Software Encryption - 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers*, volume 9054 of *Lecture Notes in Computer Science*, pages 150–171. Springer, 2015.

BN18. Srimanta Bhattacharya and Mridul Nandi. Revisiting variable output length xor pseudorandom function. *IACR Transactions on Symmetric Cryptology*, 2018(1):314–335, Mar. 2018.

BZD+16. Hanno Böck, Aaron Zauner, Sean Devlin, Juraj Somorovsky, and Philipp Jovanovic. Nonce-disrespecting adversaries: Practical forgery attacks on GCM

in TLS. In *10th USENIX Workshop on Offensive Technologies, WOOT 16, Austin, TX, USA, August 8-9, 2016*, 2016.

CHP+17. Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. A security analysis of Deoxys and its internal tweakable block ciphers. *IACR Trans. Symm. Cryptol.*, 2017(3):73–107, 2017.

CS14. Shan Chen and John Steinberger. Tight Security Bounds for Key-Alternating Ciphers. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014*, volume 8441, pages 327–350. Springer, 2014. Full version available at `http://eprint.iacr.org/2013/222`.

CS16. Benoît Cogliati and Yannick Seurin. EWCDM: an efficient, beyond-birthday secure, nonce-misuse resistant MAC. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016*, volume 9814 of *Lecture Notes in Computer Science*, pages 121–149. Springer, 2016.

DHT17. Wei Dai, Viet Tung Hoang, and Stefano Tessaro. Information-theoretic indistinguishability via the chi-squared method. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 497–523, Cham, 2017. Springer International Publishing.

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. Symm. Cryptol.*, 2018(2):161–191, 2018.

DS08. Hüseyin Demirci and Ali Aydin Selçuk. A meet-in-the-middle attack on 8-round AES. In Kaisa Nyberg, editor, *FSE 2008*, volume 5086 of *LNCS*, pages 116–126. Springer, Heidelberg, February 2008.

GK14. Shay Gueron and Michael E. Kounavis. Intel Carry-Less Multiplication Instruction and its Usage for Computing the GCM Mode. Technical report, Intel Corporation, 2014. Available at: https://software.intel.com/sites/default/files/managed/72/cc/clmul-wp-rev-2.02-2014-04-20.pdf.

GL15. Shay Gueron and Yehuda Lindell. GCM-SIV: full nonce misuse-resistant authenticated encryption at under one cycle per byte. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 109–119. ACM, 2015.

GLL17. Shay Gueron, Adam Langley, and Yehuda Lindell. AES-GCM-SIV: specification and analysis. *IACR Cryptol. ePrint Arch.*, 2017:168, 2017.

GLL19. Shay Gueron, Adam Langley, and Yehuda Lindell. AES-GCM-SIV: nonce misuse-resistant authenticated encryption. *RFC*, 8452:1–42, 2019.

Gue09. Shay Gueron. Intel's new AES instructions for enhanced performance and security (invited talk). In Orr Dunkelman, editor, *FSE 2009*, volume 5665 of *LNCS*, pages 51–66. Springer, Heidelberg, February 2009.

Hou05. R. Housley. Using advanced encryption standard (aes) ccm mode with ipsec encapsulating security payload (esp). *RFC*, 4309, 2005.

IM16. Tetsu Iwata and Kazuhiko Minematsu. Stronger security variants of GCM-SIV. *IACR Trans. Symm. Cryptol.*, 2016(1):134–157, 2016. `http://tosc.iacr.org/index.php/ToSC/article/view/539`.

IMPS17. Tetsu Iwata, Kazuhiko Minematsu, Thomas Peyrin, and Yannick Seurin. ZMAC: A fast tweakable block cipher mode for highly secure message authentication. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017*, volume 10403 of *Lecture Notes in Computer Science*, pages 34–65. Springer, 2017.

IS17. Tetsu Iwata and Yannick Seurin. Reconsidering the security bound of AES-GCM-SIV. *IACR Trans. Symm. Cryptol.*, 2017(4):240–267, 2017.

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 2014, Part II*, volume 8874 of *LNCS*, pages 274–288. Springer, Heidelberg, December 2014.

JNPS16. Jérémy Jean, Ivica Nikolic, Thomas Peyrin, and Yannick Seurin. Deoxys v1. 41. *Submitted to CAESAR*, 2016.

JNPS18. Jérémy Jean, Ivica Nikolic, Thomas Peyrin, and Yannick Seurin. Deoxys v1.43. 2018.

JNPS21. Jérémy Jean, Ivica Nikolic, Thomas Peyrin, and Yannick Seurin. The deoxys AEAD family. *J. Cryptol.*, 34(3):31, 2021.

KKS01. John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified boomerang attacks against reduced-round MARS and Serpent. In Bruce Schneier, editor, *FSE 2000*, volume 1978 of *LNCS*, pages 75–93. Springer, Heidelberg, April 2001.

KLW17. Thorsten Kranz, Gregor Leander, and Friedrich Wiemer. Linear cryptanalysis: Key schedules and tweakable block ciphers. *IACR Trans. Symm. Cryptol.*, 2017(1):474–505, 2017.

LJ19. Rongjia Li and Chenhui Jin. Meet-in-the-middle attacks on round-reduced tweakable block cipher deoxys-bc. *IET Inf. Secur.*, 13(1):70–75, 2019.

LN17. Eik List and Mridul Nandi. Revisiting full-PRF-secure PMAC and using it for beyond-birthday authenticated encryption. In Helena Handschuh, editor, *CT-RSA 2017*, volume 10159 of *LNCS*, pages 258–274. Springer, Heidelberg, February 2017.

LRW02. Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 31–46. Springer, Heidelberg, August 2002.

LSG+20. Ya Liu, Bing Shi, Dawu Gu, Fengyu Zhao, Wei Li, and Zhiqiang Liu. Improved meet-in-the-middle attacks on reduced-round deoxys-bc-256. *Comput. J.*, 63(12):1859–1870, 2020.

MMS18. Farokhlagha Moazami, Alireza Mehrdad, and Hadi Soleimany. Impossible differential cryptanalysis on deoxys-bc-256. *ISC Int. J. Inf. Secur.*, 10(2):93–105, 2018.

MN17a. Bart Mennink and Samuel Neves. Encrypted davies-meyer and its dual: Towards optimal security using mirror theory. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017, Proceedings, Part III*, volume 10403 of *LNCS*, pages 556–583. Springer, 2017.

MN17b. Bart Mennink and Samuel Neves. Optimal PRFs from blockcipher designs. *IACR Trans. Symm. Cryptol.*, 2017(3):228–252, 2017.

MV04. David A. McGrew and John Viega. The security and performance of the Galois/counter mode (GCM) of operation. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT 2004*, volume 3348 of *LNCS*, pages 343–355. Springer, Heidelberg, December 2004.

NIS07. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) andGMAC. NIST Special Publication 800-38D, 2007.

Pat08. Jacques Patarin. The "Coefficients H" Technique. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography - SAC 2008*, volume 5381, pages 328–345. Springer, 2008.

PS16. Thomas Peyrin and Yannick Seurin. Counter-in-tweak: Authenticated encryption modes for tweakable block ciphers. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 33–63. Springer, Heidelberg, August 2016.

RBBK01. Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In Michael K. Reiter and Pierangela Samarati, editors, *ACM CCS 2001*, pages 196–205. ACM Press, November 2001.

RS06. Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004, pages 373–390. Springer, 2006.

VP. Mathy Vanhoef and Frank Piessens. Key reinstallation attacks: Forcing nonce reuse in WPA2. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017*, pages 1313–1328.

ZDJ19. Boxin Zhao, Xiaoyang Dong, and Keting Jia. New related-tweakey boomerang and rectangle attacks on deoxys-bc including BDT effect. *IACR Trans. Symm. Cryptol.*, 2019(3):121–151, 2019.

Supplementary Material

A Proof of Theorem 1

The first step of the proof is to compute the value of $\chi^2(s^{i-1})$ for all $i = 1, \ldots, q$ and all $s^{i-1} \in \Omega^{i-1}$. Let us fix any integer i in $\{1, \ldots, q\}$, and any transcript s^{i-1} such that $\Pr_{\mathsf{S}}\left[s^{i-1}\right] > 0$. Recall that one has

$$\chi^2(s^{i-1}) = \sum_{s \in \Omega_{s^{i-1}}} \frac{\left(\Pr_{\mathsf{S}}\left[s_i | s^{i-1}\right] - \Pr_{\mathsf{R}}\left[s_i | s^{i-1}\right]\right)^2}{\Pr_{\mathsf{R}}\left[s_i | s^{i-1}\right]},$$

where $\Omega_{s^{i-1}} = \{s_i : s^i \in \Omega_i\}$. Let us fix any $s_i \in \Omega_{s^{i-1}}$. The value of s^{i-1} completely determines the value of the i-th adversarial query, which will be denoted (t_i, x_i). Let n_i be the number of occurrences of the tweak t_i in s^{i-1}. Clearly, one has $n_i \leq i - 1$. Besides, in the system S, the output value $(y_{1,i} \| \cdots \| y_{\alpha,i}, z_{0,i})$ uniquely determines the value of the random variables $Z_{j,i}$ for $j = 1, \ldots, \alpha$, and these values are compatible with the output of a tweakable permutation since $s_i \in \Omega_{s^{i-1}}$. It is easy to see that, at this point, the random variable Z_j in Algorithm 1 is sampled uniformly at random in a set of size $2^n - n_i$ for $j = 0, \ldots, \alpha$. Hence, one has

$$\Pr_{\mathsf{S}}\left[s_i | s^{i-1}\right] = \frac{1}{(2^n - n_i)^{\alpha+1}}.$$

Let us denote $D_{s^{i-1},j}$ the set of all $z_{0,k} \oplus y_{j,k}$ such that $t_k = t_i$, for $k = 1, \ldots, i-1$ and $j = 1, \ldots, \alpha$. Similarly, let $D_{s^{i-1},0}$ be the set of all $z_{0,k}$ such that $t_k = t_i$, for $k = 1, \ldots, i-1$. Namely, these sets correspond to the value of $D_j[T_i]$ at the beginning of the i-th query in both experiments. In the ideal world, the sampling of the output is done as follows:

- first $Y_i = Y_{1,i} \| \cdots \| Y_{\alpha,i}$ is chosen uniformly at random in $\{0,1\}^{\alpha n}$;
- second, $Z_{0,i}$ is chosen uniformly at random outside of $D^{y_i}_{s^{i-1}} := D_{s^{i-1},0} \cup \{y_{j,i} \oplus d_j : j = 1, \ldots, \alpha, d_j \in D_{s^{i-1},j}\}$.

Hence, one clearly has

$$\Pr_{\mathsf{R}}\left[s_i | s^{i-1}\right] = \frac{1}{2^{\alpha n}\left(2^n - \left|D^{y_i}_{s^{i-1}}\right|\right)}.$$

Thus

$$\begin{aligned}
\chi^2(s^{i-1}) &= \sum_{s\in\Omega_{s^{i-1}}} 2^{\alpha n}\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right)\left(\frac{1}{(2^n-n_i)^{\alpha+1}} - \frac{1}{2^{\alpha n}\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right)}\right)^2 \\
&= \sum_{s\in\Omega_{s^{i-1}}} 2^{\alpha n}\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right)\left(\frac{\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right) - \frac{(2^n-n_i)^{\alpha+1}}{2^{\alpha n}}}{(2^n-n_i)^{\alpha+1}\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right)}\right)^2 \\
&= \sum_{(t_i,x_i,y_i,z_{0,i})\in\Omega_{s^{i-1}}} \frac{2^{\alpha n}\left(\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right) - \frac{(2^n-n_i)^{\alpha+1}}{2^{\alpha n}}\right)^2}{(2^n-n_i)^{2\alpha+2}\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right)} \\
&= \sum_{y_i\in\{0,1\}^{\alpha n}} \frac{2^{\alpha n}\left(\left(2^n - \left|D_{s^{i-1}}^{y_i}\right|\right) - \frac{(2^n-n_i)^{\alpha+1}}{2^{\alpha n}}\right)^2}{(2^n-n_i)^{2\alpha+2}}. \qquad (2)
\end{aligned}$$

The remaining step is to compute the expectancy of $\chi^2(s^{i-1})$ when s^{i-1} is sampled according to S. Note that this random experiment can be uniquely rewritten as a sampling of the Z random variables, as per Algorithm 1, which will make our computation slightly easier to read. Let us start by remarking that $2^n - \left|D_{s^{i-1}}^{y_i}\right| = \left|\overline{D_{s^{i-1}}^{y_i}}\right|$, where $\overline{A}$ denotes the complementary of the set A in $\{0,1\}^{\alpha n}$. Besides, one also has

$$\left|\overline{D_{s^{i-1}}^{y_i}}\right| = \sum_{z\in\{0,1\}^n} \mathsf{I}_z,$$

where $\mathsf{I}_z = 1$ if $z \in \overline{D_{s^{i-1}}^{y_i}}$, and 0 otherwise. In the real world, $\mathsf{I}_z = 1$ corresponds to the following event:

- the random values $Z_{0,k}$, for $k = 1,\ldots,n_i$, which are sampled without replacement in $\{0,1\}^n$, are all different from z;
- for $j = 1,\ldots,\alpha$ and $k = 1,\ldots,n_i$, the random values $Z_{j,k}$, which are sampled without replacement in $\{0,1\}^n$, are all different from $z \oplus y_{j,i}$.

Hence, one has

$$\mathbb{E}\left[I_z\right] = \Pr\left[\mathsf{I}_z = 1\right] = \frac{(2^n-1)_{n_i}^{\alpha+1}}{(2^n)_{n_i}^{\alpha+1}} = \left(\frac{2^n-n_i}{2^n}\right)^{\alpha+1},$$

and

$$\mathbb{E}\left[\left|\overline{D_{s^{i-1}}^{y_i}}\right|\right] = \sum_{z\in\{0,1\}^n} \mathbb{E}\left[\mathsf{I}_z\right] = \frac{(2^n-n_i)^{\alpha+1}}{2^{\alpha n}}.$$

Plugging this equality into Eq (2), one has

$$\mathbb{E}\left[\chi^2(S^{i-1})\right] = \sum_{y_i \in \{0,1\}^{\alpha n}} \frac{2^{\alpha n}\mathbb{E}\left[\left(\left|\overline{D^{y_i}_{S^{i-1}}}\right| - \mathbb{E}\left[\left|\overline{D^{y_i}_{S^{i-1}}}\right|\right]\right)^2\right]}{(2^n - n_i)^{2\alpha+2}}$$
$$= \sum_{y_i \in \{0,1\}^{\alpha n}} \frac{2^{\alpha n}\mathrm{Var}\left[\left|\overline{D^{y_i}_{S^{i-1}}}\right|\right]}{(2^n - n_i)^{2\alpha+2}}. \tag{3}$$

This implies that we now have to evaluate $\mathrm{Var}\left[\left|\overline{D^{y_i}_{S^{i-1}}}\right|\right]$ for every $y_i \in \{0,1\}^{\alpha n}$. One has

$$\begin{aligned}
\mathrm{Var}\left[\left|\overline{D^{y_i}_{S^{i-1}}}\right|\right] &= \mathrm{Var}\left[\sum_{z \in \{0,1\}^n} \mathsf{I}_z\right] \\
&= \sum_{z \in \{0,1\}^n} \mathrm{Var}\left[\mathsf{I}_z\right] + \sum_{z \neq z'} \mathrm{Covar}\left(\mathsf{I}_z, \mathsf{I}_{z'}\right) \\
&= \sum_{z \in \{0,1\}^n} \left(\mathbb{E}\left[\mathsf{I}_z^2\right] - \mathbb{E}\left[\mathsf{I}_z\right]^2\right) + \sum_{z \neq z'} \left(\mathbb{E}\left[\mathsf{I}_z\mathsf{I}_{z'}\right] - \mathbb{E}\left[I_z\right]\mathbb{E}\left[I_{\prime}\right]\right) \\
&= \sum_{z \in \{0,1\}^n} \left(\mathbb{E}\left[\mathsf{I}_z\right] - \mathbb{E}\left[\mathsf{I}_z\right]^2\right) + \sum_{z \neq z'} \left(\mathbb{E}\left[\mathsf{I}_z\mathsf{I}_{z'}\right] - \mathbb{E}\left[I_z\right]^2\right) \\
&= \sum_{z \in \{0,1\}^n} \left(\frac{2^n - n_i}{2^n}\right)^{\alpha+1} \left(1 - \left(\frac{2^n - n_i}{2^n}\right)^{\alpha+1}\right) + \sum_{z \neq z'} \left(\mathbb{E}\left[\mathsf{I}_z\mathsf{I}_{z'}\right] - \mathbb{E}\left[I_z\right]^2\right).
\end{aligned}$$

As was done previously for the computation of $\mathbb{E}\left[\mathsf{I}_z\right]$, $\mathsf{I}_z\mathsf{I}_{z'} = 1$ corresponds to the following event:

- the random values $Z_{0,k}$, for $k = 1, \ldots, n_i$, which are sampled without replacement in $\{0,1\}^n$, are all different from z and z', with $z \neq z'$;
- for $j = 1, \ldots, \alpha$ and $k = 1, \ldots, n_i$, the random values $Z_{j,k}$, which are sampled without replacement in $\{0,1\}^n$, are all different from $z \oplus y_{j,i}$ and $z' \oplus y_{j,i}$.

Thus, one has

$$\mathbb{E}\left[\mathsf{I}_z\mathsf{I}_{z'}\right] = \Pr\left[\mathsf{I}_z\mathsf{I}_{z'} = 1\right] = \frac{(2^n - 2)^{\alpha+1}_{n_i}}{(2^n)^{\alpha+1}_{n_i}} = \left(\frac{(2^n - n_i)(2^n - n_i - 1)}{2^n(2^n - 1)}\right)^{\alpha+1}$$
$$= \left(1 - \frac{n_i}{2^n}\right)^{\alpha+1} \left(1 - \frac{n_i}{2^n - 1}\right)^{\alpha+1} \leq \left(1 - \frac{n_i}{2^n}\right)^{2\alpha+2} = \mathbb{E}\left[I_z\right]^2.$$

Thus, it is easy to see that one has

$$\begin{aligned}\mathrm{Var}\left[\left|\overline{D^{y_i}_{S^{i-1}}}\right|\right] &\leq \sum_{z\in\{0,1\}^n}\left(\frac{2^n-n_i}{2^n}\right)^{\alpha+1}\left(1-\left(\frac{2^n-n_i}{2^n}\right)^{\alpha+1}\right)\\ &\leq \frac{(2^n-n_i)^{\alpha+1}}{2^{\alpha n}}\left(1-\left(1-\frac{n_i}{2^n}\right)^{\alpha+1}\right)\\ &\leq \frac{(2^n-n_i)^{\alpha+1}(\alpha+1)n_i}{2^{(\alpha+1)n}}. \end{aligned} \tag{4}$$

Using Lemma 2, Eqs (3) and (4), one has

$$\begin{aligned}\|\mathrm{Pr}_{\mathsf{S}}-\mathrm{Pr}_{\mathsf{R}}\|^2 &\leq \frac{1}{2}\sum_{i=1}^{q}\mathbb{E}\left[\chi^2(\mathsf{S}^{i-1})\right]\\ &\leq \frac{1}{2}\sum_{i=1}^{q}\sum_{y_i\in\{0,1\}^{\alpha n}}\frac{2^{\alpha n}\mathrm{Var}\left[\left|\overline{D^{y_i}_{S^{i-1}}}\right|\right]}{(2^n-n_i)^{2\alpha+2}}\\ &\leq \frac{1}{2}\sum_{i=1}^{q}\sum_{y_i\in\{0,1\}^{\alpha n}}\frac{2^{\alpha n}(2^n-n_i)^{\alpha+1}(\alpha+1)n_i}{(2^n-n_i)^{2\alpha+2}2^{(\alpha+1)n}}\\ &\leq \frac{1}{2}\sum_{i=1}^{q}\sum_{y_i\in\{0,1\}^{\alpha n}}\frac{(\alpha+1)n_i}{(2^n-n_i)^{\alpha+1}2^n}\\ &\leq \frac{1}{2}\sum_{i=1}^{q}\frac{2^{\alpha n}(\alpha+1)(i-1)}{(2^n-q)^{\alpha+1}2^n} \leq \frac{1}{4}\frac{2^{(\alpha-1).n}(\alpha+1)q^2}{(2^n-q)^{\alpha+1}}. \end{aligned}$$

Since one has $(\alpha+1)q \leq 2^n$ by assumption, $2^n - q \geq 2^n\frac{\alpha}{\alpha+1}$. This implies that

$$\|\mathrm{Pr}_{\mathsf{S}}-\mathrm{Pr}_{\mathsf{R}}\|^2 \leq \frac{1}{4}\frac{(\alpha+1)q^2}{2^{2n}}\left(\frac{\alpha+1}{\alpha}\right)^{\alpha+1}.$$

Moreover, one has

$$\ln\left(\frac{\alpha+1}{\alpha}\right) = \ln(\alpha+1)-\ln(\alpha) = \int_{\alpha}^{\alpha+1}\frac{1}{t}\mathsf{d}t \leq \frac{1}{\alpha}.$$

Thus $\left(\frac{\alpha+1}{\alpha}\right)^{\alpha+1} \leq \exp((\alpha+1)\alpha^{-1}) \leq \exp(2) \leq 8$ since $\alpha \geq 1$.

B Graphical Representation of SAFE

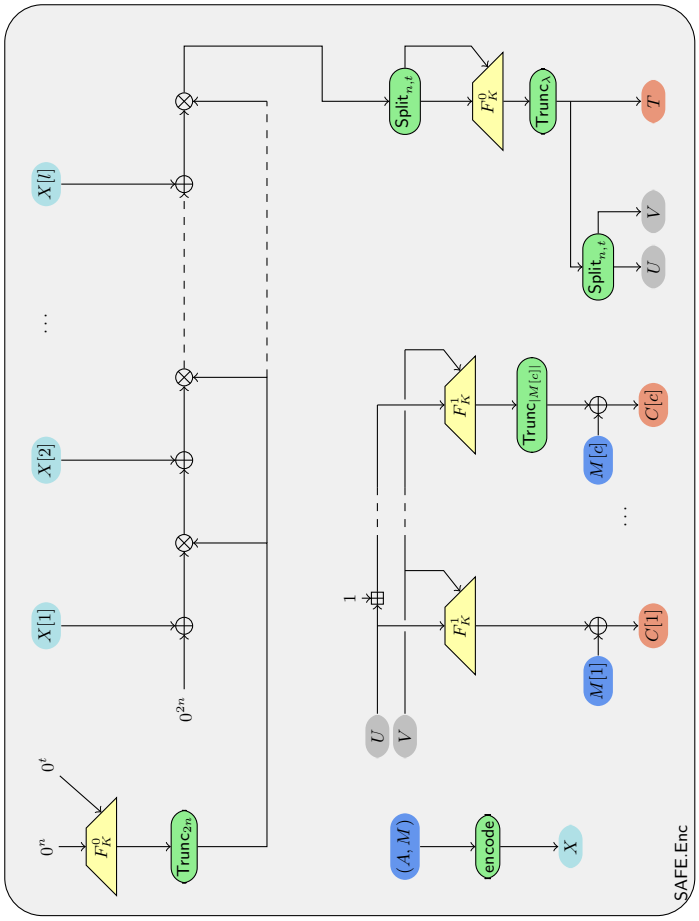

Fig. 7: The SAFE authenticated encryption function which, given a plaintext M and associated data A, outputs the ciphertext C and the tag T. Here Trunc_i simply truncates its input to i bits if it is longer or fills it with zeroes otherwise, and $\mathsf{Split}_{a,b}$ additionally splits it into its leftmost a bits and rightmost b bits. $\otimes$ denotes the multiplication in $\mathrm{GF}(2^{2n})$. Recall that $\lambda \leq 2n$.

C ZMAC Pseudocode

Algorithm 5 Specification of ZMAC [IMPS17]. Here $2a$ denotes the multiplication by the element of the finite field $\mathrm{GF}(2^n)$ that is represented by the polynomial x.

```
function ZHASH(K,X)
    U ← 0^n, V ← 0^t
    L_l ← E^9_K(0^t, 0^n)
    L_r ← E^9_K(0^{t-1}1, 0^n)
    (X[1], ..., X[m]) ←^{n+t}— X
    for i = 1 to m do
        (X_l, X_r) ←^{n,t}— X[i]
        S_l ← L_l ⊕ X_l
        S_r ← L_r ⊕_t X_r
        C_l ← E^8_K(S_r, S_l)
        C_r ← C_l ⊕_t X_r
        U ← 2(U ⊕ C_l)
        V ← V ⊕ C_r
        (L_l, L_l) ← (2L_l, 2L_l)
    end for
    return (U, V)
end function
```

```
function ZFIN(K,i,U,V)
    Y[1] ⟵ E^i_K(V,U) ⊕ E^{i+1}_K(V,U)
    Y[2] ⟵ E^{i+2}_K(V,U) ⊕ E^{i+3}_K(V,U)
    Y ⟵ Y[1]||Y[2]
    return Y
end function
```

```
function ZMAC(K,M)
    X ⟵ Pad_10(M)
    (U,V) ⟵ ZHASH(K,X)
    if n + t | |M| then
        return ZFIN(K,0,U,V)
    else
        return ZFIN(K,4,U,V)
    end if
end function
```

D Deferred Proofs from Section 6

D.1 On the SIV generic construction

The generic SIV construction, as defined in [RS06], takes the composition of a PRF and an IV-based encryption scheme, where the output of the PRF is used as the IV of the encryption scheme, in order to create a DAE scheme. In our context, while we use the full output of the PRF as tag, we truncate it before using it as an IV. This requires a slight adjustment to [RS06, Theorem 1], which we develop in this section.

Let $N \geq M$ be two positive integers, let $F : \mathcal{K}_1 \times \mathcal{AD} \times \mathcal{M} \to \{0,1\}^N$ be a PRF, and let E be an IV-based encryption scheme with message space $\mathcal{M}$, ciphertext space $\mathcal{C}$, key space $\mathcal{K}_2$, and IV space $\{0,1\}^M$.

The DAE scheme $\mathsf{SIV}[F,E]$ is defined in algorithm 6. Note that this corresponds exactly to the composition method that we use for SAFE, where $N = \lambda$, and $M = \min(\lambda, n+t)$. One has the following result.

Theorem 4. *Let D be an adversary against the* dae*-security of $\mathsf{SIV}[F,E]$ that runs in time t and makes q queries of total length σ. Then there exist an adversary D_1 against the* prf*-security of F and an adversary D_2 against the* ive*-security of E such that*

$$\mathrm{Adv}^{\mathrm{dae}}_{\mathsf{SIV}[F,E]}(D) \leq \mathrm{Adv}^{\mathrm{prf}}_{F}(D_1) + \mathrm{Adv}^{\mathrm{ive}}_{E}(D_2) + \frac{q}{2^N}.$$

Algorithm 6 The $\mathsf{SIV}[F, E]$ Authenticated Encryption scheme, where F is a PRF that outputs N-bit strings, and E is an IV-based encryption scheme that accepts IVs of length $M \leq N$ bits.

```
function SIV[F,E].Enc((K1,K2),A,X)
    T ⟵ F(K1,A,X)
    IV ⟵ [T]_M
    C ⟵ E.Enc(K2,IV,X)
    return (C,T)
end function
```

```
function SIV[F,E].Dec((K1,K2),A,C,T)
    IV ⟵ [T]_M
    X ⟵ E.Dec(K2,IV,C)
    T' ⟵ F(K1,A,X)
    if T = T' then
        return X
    else
        return ⊥
    end if
end function
```

The adversaries D_1 and D_2 run in time at most $t + c\sigma$ for some constant c, and make at most q queries of total length at most σ.

The proof of this result is almost exactly the same as the one of [RS06, Theorem 1], and we state it here for the sake of completeness.

Proof. Let D be an adversary against the dae-security of $\Pi := \mathsf{SIV}[F, E]$ that runs in time t and makes q queries of total length σ. We also assume w.l.o.g that D never repeats queries. We are going to introduce an intermediary DAE scheme $\Pi' := \mathsf{SIV}[\$, E]$ which is similar to $\mathsf{SIV}[F, E]$, but where F has been replaced by a truly random function. Then, the following inequalities hold:

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{dae}}_{\mathsf{SIV}[F,E]}(D) &= \left|\Pr\left[D^{\Pi.\mathsf{Enc}_K, \Pi.\mathsf{Dec}_K} = 1\right] - \Pr\left[D^{\$(\cdot,\cdot),\perp(\cdot,\cdot)} = 1\right]\right| \\
&\leq \left|\Pr\left[D^{\Pi.\mathsf{Enc}_K, \Pi.\mathsf{Dec}_K} = 1\right] - \Pr\left[D^{\Pi'.\mathsf{Enc}_{K_2}, \Pi'.\mathsf{Dec}_{K_2}} = 1\right]\right| \\
&\qquad + \left|\Pr\left[D^{\Pi'.\mathsf{Enc}_{K_2}, \Pi'.\mathsf{Dec}_{K_2}} = 1\right] - \Pr\left[D^{\Pi'.\mathsf{Enc}_{K_2}, \perp(\cdot,\cdot)}\right]\right| \\
&\qquad + \left|\Pr\left[D^{\Pi'.\mathsf{Enc}_{K_2}, \perp(\cdot,\cdot)}\right] - \Pr\left[D^{\$(\cdot,\cdot),\perp(\cdot,\cdot)}\right]\right| \qquad (5) \\
&= p_1 + p_2 + p_3, \qquad (6)
\end{aligned}$$

where p_i denotes the i-th term of the r.h.s. of the inequality.

It is easy to see that p_1 corresponds to the advantage of the following adversary D_1 against the prf-security of F: D_1 picks a uniformly random key K_2 in $\mathcal{K}_2$ and then runs D, and outputs the same bit. It simply forwards D's encryption queries to its oracle in order to generate the tag, then simulates the encryption of the plaintext using its own key K_2 and the truncated tag to generate the ciphertext. For every decryption query, it simply decrypts the ciphertext, and feeds the output to its oracle to generate the tag, and returns the plaintext if both tags match, or $\perp$ otherwise. It is clear that D_1 makes at most q queries, for a total length of at most σ, and runs in time $t + c\sigma$, where c is a small constant depending on the overhead of evaluating E. Hence, one clearly has

$$p_1 = \mathrm{Adv}^{\mathrm{prf}}_F(D_1). \qquad (7)$$

Similarly, p_3 corresponds to the ive-advantage of the following adversary D_2 against the security of E. Here D_2 also runs D and outputs the same bit. Let us generically denote with g the oracle D_1 has access to. For every encryption query (A, X), it simply asks for the encryption $(\mathrm{IV}, C) = g(X)$ of the queried plaintext, then chooses uniformly at random a $N-M$-bit string T', then forwards $(\mathrm{IV}||T', C)$ as the (tag, ciphertext) pair. For every decryption query, it simply answers $\perp$. If $g = E^{\$}_{K_2}.\mathsf{Enc}$, then D_1 perfectly simulates $\Pi'.\mathsf{Enc}_{K_2}$ since the tag is indeed uniformly random, and the ciphertext can be generated by applying $E_{K_2}..\mathsf{Enc}$ to the truncated tag and the plaintext. Note that here, the argument relies on the fact that D never repeats queries. If g outputs uniformly random strings, then clearly D_1 perfectly simulates $\$(\cdot,\cdot)$. Hence, one has

$$p_3 \leq \mathrm{Adv}^{\mathrm{ive}}_{E}(D_2), \tag{8}$$

where D_2 makes at most q queries for a total length of at most σ, and runs in time at most $t + c\sigma$.

In order to upper bound p_2, we are first going to slightly alter the security experiment. We are going to reveal the key K_2 to the attacker. Since this can only improve its advantage, this can be done w.l.o.g. We can also assume that D outputs 1 as soon as a decryption query succeeds, since the encryption oracles are the same in both worlds, which means that encryption query cannot help for distinguishing between the two pairs of oracles. If we take a look at algorithm 6, this means that p_2 is upper bounded by the probability that D makes a decryption query (A, C, T) such that $f(A, X) = T$, where f denotes the uniformly random function that underlies Π', and $X = E.\mathsf{Dec}_{K_2}([T]_M, C)$. Since the adversary cannot make trivial queries, this equality occurs with a probability smaller than 2^{-n}, which means that one has

$$p_2 \leq \frac{q}{2^N}. \tag{9}$$

Combining Equation (5) with Eqs (7), (8) and (9) yields the result. □

D.2 Proof of Theorem 2

We are going to fix four positive integers n, t, m, λ such that $n \leq m$ and $\lambda \leq 2n$ for the whole proof of Theorem 2. Let F be a TPRF with key space K, such that for every K in $\mathcal{K}$, one has $F_K \in \mathrm{Func}(t+1, n, m)$.

Let also q, σ be two integers. Let A be an adversary against the DAE-security of $\mathsf{SAFE}[F]$ that makes at most q queries, for a total of at most σ n-bit blocks, and runs in time at most $\mathtt{time}$.

As usual, the first step of the construction is to use a hybrid argument to replace the TPRF F by a uniformly random tweakable function F_0. We then denote by SAFE_0, SFMac_0 and FEnc_0 the resulting constructions. More specifically, A can be turned into an attacker B against the tprf- security of F that makes at most $\sigma + q + 1$ chosen plaintext queries and runs in time $\mathtt{time} + \alpha\sigma$, where α is an upper bound on the overhead implied by SAFE. Then, one has

$$\mathrm{Adv}^{\mathrm{dae}}_{\mathsf{SAFE}[F]}(A) \leq \mathrm{Adv}^{\mathrm{tprf}}_{F}(B) + \mathrm{Adv}^{\mathrm{dae}}_{\mathsf{SAFE}_0}(A). \tag{10}$$

Then, what remains is to upper bound the advantage of A against the dae-security of SAFE. Since this experiment is purely information-theoretical, we can assume w.l.o.g. that A is computationally unbounded, and thus deterministic, and does not make useless queries. Applying a small variant of [RS06, Theorem 1] that we discuss in more details in Supplementary Material D.1, we know that there exist two information-theoretical adversaries C (resp. D) against the prf-security of SFMac (resp. the ive-security of FEnc) that both make at most q queries for a total of at most σ n-bit blocks, such that

$$\mathrm{Adv}^{\mathrm{dae}}_{\mathsf{SAFE}_0}(A) \leq \frac{q}{2^\lambda} + \mathrm{Adv}^{\mathrm{prf}}_{\mathsf{SFMac}_0}(C) + \mathrm{Adv}^{\mathrm{ive}}_{\mathsf{FEnc}_0}(D). \tag{11}$$

Note that this strategy works thanks to the domain separation of the TPRF instances.

One has the following Theorem on the prf-security of SFMac_0.

Theorem 5. *Let $n, t, m, q, \sigma, \lambda$ be positive integers such that $m \geq 2n \geq 1$, $q, \sigma < 2^n$ and $\lambda \leq 2n$. Let D be a computationally unbounded deterministic adversary against the* prf*-security of SFMac_0 that makes at most q queries for a total of at most σ n-bit blocks. Then one has*

$$\mathrm{Adv}^{\mathrm{prf}}_{\mathsf{SFMac}_0}(D) \leq \frac{2q\sigma + 2q^2 + \sigma + 4q}{2^{n+\min(n,t)}}.$$

Proof sketch. Since the finalization function of SFMac_0 is a uniformly random function, as long as there is no hash collision and that no hash is all-zero, then the outputs will be indistinguishable from uniform. This probability can be shown to be upper-bounded by

$$\frac{2q\sigma + 2q^2 + \sigma + 4q}{2^{n+\min(n,t)}}.$$

A complete proof of Theorem 5 can be found in Supplementary Material D.3. Besides, we also prove the following result on the ive-security of FEnc_0.

Theorem 6. *Let $n, t, m, q, \sigma, \lambda$ be positive integers such that $m \geq n$, $q, \sigma < 2^n$ and $\lambda \leq 2n$. Let us define $m' = \min(\lambda, n + t)$. Let D be a computationally unbounded deterministic adversary against the* ive*-security of FEnc_0 that makes at most q queries for a total of at most σ m-bit blocks, and never repeats a query. Then one has*

$$\mathrm{Adv}^{\mathrm{ive}}_{\mathsf{FEnc}_0}(D) \leq \frac{2(q-1)\sigma}{2^{m'}}.$$

We are going to use the H coefficients technique to prove this Theorem. Let q, σ be two integers such that $q, \sigma < 2^n$. Let D be a computationally unbounded deterministic adversary against the ive-security of FEnc_0, as stated in the theorem. We make the additional assumption that D always queries the maximum number of blocks σ. Then D has to distinguish between two worlds:

- the real world where it interacts with the randomized construction FEnc_0^*;
- the ideal world where it interacts with an oracle that on input M outputs a uniformly random string of size $\{0,1\}^{m'+|M|}$.

As before, the interaction of D with its oracle will be summarized in a queries transcript $\mathcal{Q}$ of the attack. It will consist in the list $\mathcal{Q} = \{(I_1, M_1, C_1), \ldots, (I_q, M_q, C_q)\}$, where, for $i = 1, \ldots, q$, D queried message M_i, and received IV I_i, and ciphertext C_i. Moreover, we will denote by m_i the length of M_i in m-bit blocks, $I_{n,i}||I_{t,i} \xleftarrow{(n,t)} [I_i]_{n+t}$ and

$$M_i[1]|| \cdots ||M_i[m_i] \xleftarrow{m} M_i,$$
$$C_i[1]|| \cdots ||C_i[m_i] \xleftarrow{m} C_i.$$

We are going to slightly alter the security experiment by always outputting the full output of the last m-bit keystream block in the real world, regardless of the actual size of the input. In the ideal world, we simply output additional random bits. As D can freely ignore it, this can only improve its advantage.

The next step in our proof will be to define a set of bad transcripts and upper bound their probability of appearing in the ideal word.

Definition 6. *An attainable transcript $\mathcal{Q}$ will be said* bad *if there exists four integers i, i', j, j' such that $1 \leq i < j \leq q$, $1 \leq i' \leq m_i$, $1 \leq j' \leq m_j$, and*

$$I_{t,i} = I_{t,j}, I_{n,i} \boxplus (i' - 1) = I_{n,j} \boxplus (j' - 1).$$

One has the following result.

Lemma 4. *One has* $\Pr[\theta_{\text{id}} \in \Theta_{\text{bad}}] \leq \frac{2(q-1)\sigma}{2^{m'}}$.

The proof of this Lemma has been deferred to Supplementary Material D.6. Let us now consider good transcripts. One has the following result.

Lemma 5. *Let $\mathcal{Q}$ be a good queries transcript. One has*

$$\Pr[\theta_{\text{re}} = \tau] = \Pr[\theta_{\text{id}} = \tau].$$

The proof of this Lemma has been deferred to Supplementary Material D.7. Combining Lemmas 1 with Lemmas 4 and 5 ends the proof of Theorem 6, while combining Lemmas 1, 6 and 7 ends the proof of Theorem 5.

D.3 Proof of Theorem 5

We are going to use the H coefficients technique to prove this Theorem. Let q, σ be two integers such that $q, \sigma < 2^n$. Since the result is trivially true when $q = 0$ or $\sigma = 0$, we are going to assume that $q, \sigma > 0$. Let D be a computationally unbounded deterministic adversary against the prf-security of SFMac_0, as stated in the theorem. Then D has to distinguish between two worlds:

- the real world where it interacts with the construction SFMac_0;
- the ideal world where it interacts with a uniformly random function with domain $(\{0,1\}^*)^2$ and range $\{0,1\}^\lambda$.

As usual, the interaction of D with its oracle will be summarized in a queries transcript $\mathcal{Q}$ of the attack. It will consist in the list $\mathcal{Q} = \{(A_1, M_1, T_1), \ldots, (A_q, M_q, T_q)\}$, where, for $i = 1, \ldots, q$, D queried message M_i with associated data A_i, and received tag T_i. Moreover, we will denote by x_i the length of the encoding of (A_i, M_i) in $2n$-bit blocks, and

$$X_i[1]|| \cdots ||X_i[x_i] \xleftarrow{2n} \mathrm{Pad}_{10}A||\mathrm{Pad}_{10}M||\langle|A|\rangle_n||\langle|M|\rangle_n.$$

Note that, if a_i (resp. m_i) denote the length of A_i (resp. M_i) in n-bit blocks, then $x_i \leq (a_i + m_i)/2 + 4$.

We are also going to reveal additional information to the adversary at the end of its interaction with its oracles, but before it outputs its answer. As D can freely ignore it, this can only improve its advantage. Namely, in the real world, we are going to reveal the value of the hashing key L, while in the ideal world we will simply draw a uniformly random value L in $\{0,1\}^{2n}$. Overall, the transcript τ of the attack will be defined as $\tau = (\mathcal{Q}, L)$. In order to simplify the description of bad transcript, we are going to denote by $H_L(A, M)$ the result of the evaluation of the hashing algorithm on (A, M) (l. 4 to 8 in 3). Note that one has

$$H_L(A_i, M_i) = \bigoplus_{i=j}^{x_i} L^{x_i - j + 1} \otimes X_i[j].$$

The next step in our proof will be to define a set of bad transcripts and upper bound their probability of appearing in the ideal word.

Definition 7. *An attainable transcript $\tau = (\mathcal{Q}, L)$ will be said* bad *if one of the following conditions hold:*

(C-1) there exist $i < j$ such that $[H_L(A_i, M_i)]_{n+t} = [H_L(A_j, M_j)]_{n+t}$;
(C-2) there exists i such that $[H_L(A_i, M_i)]_{n+t} = 0$.

One has the following result.

Lemma 6. *One has*

$$\Pr\left[\theta_{\mathrm{id}} \in \Theta_{\mathrm{bad}}\right] \leq \frac{2q\sigma + 2q^2 + \sigma + 4q}{2^{n+\min(n,t)}}.$$

The proof of this Lemma has been deferred to Supplementary Material D.4. We now consider good transcripts, and prove that the probabilities of observing any such transcript in the real world and in the ideal world are close. More precisely, the following result holds.

Lemma 7. *For any τ in Θ_{good}, one has*

$$\frac{\Pr\left[\theta_{\mathrm{re}} = \tau\right]}{\Pr\left[\theta_{\mathrm{id}} = \tau\right]} = 1.$$

The proof of this Lemma has been deferred to Supplementary Material D.5.

D.4 Proof of Lemma 6

We denote by Θ_i the set of all attainable transcripts that fulfill condition (C-i) for $i = 1, 2$. As usual, using the union bound, one has

$$\Pr\left[\theta_{\text{id}} \in \Theta_{\text{bad}}\right] \leq \sum_{i=1}^{2} \Pr\left[\theta_{\text{id}} \in \Theta_i\right]. \tag{12}$$

We are going to upper bound both probabilities in turn.

Condition (C-1). Let us fix $i < j$. The condition $[H_L(A_i, M_i)]_{n+t} = [H_L(A_j, M_j)]_{n+t}$ can be rephrased as follows:

- if $t < n$: there exists $\delta \in \{0,1\}^{n-t}$ such that $H_L(A_i, M_i) \oplus H_L(A_j, M_j) = \delta || 0^{n+t}$;
- if $t \geq n$: $H_L(A_i, M_i) \oplus H_L(A_j, M_j) = 0^{2n}$.

In both cases, the equation corresponds to a polynomial of degree at most $\max(a_i + m_i, a_j + m_j) + 4$ in L, since our encoding is injective and the adversary is not allowed to repeat queries. Since in the ideal world, L is chosen uniformly at random at the end of the queries, independently from the experiment, we can see that, in both cases, one has

$$\Pr\left[[H_L(A_i, M_i)]_{n+t} = [H_L(A_j, M_j)]_{n+t}\right] \leq \frac{a_i + a_j + m_i + m_j + 4}{2^{n+\min(n,t)}}.$$

Summing over every possible pair of queries yields

$$\Pr\left[\theta_{\text{id}} \in \Theta_1\right] \leq \frac{2q\sigma + 2q^2}{2^{n+\min(n,t)}}. \tag{13}$$

Condition (C-2). This condition can be treated like the previous one. Indeed, if we fix $1 \leq i \leq q$, then the condition $[H_L(A_i, M_i)]_{n+t} = 0$ yields a polynomial in L of degree at most x_i. Hence, one has

$$\Pr\left[[H_L(A_i, M_i)]_{n+t} = 0\right] \leq \frac{a_i + m_i + 4}{2^{n+\min(n,t)}},$$

and summing over the q queries gives

$$\Pr\left[\theta_{\text{id}} \in \Theta_2\right] \leq \frac{\sigma + 4q}{2^{n+\min(n,t)}}. \tag{14}$$

Combining Eqs. (12), (13) and (14) concludes this proof. □

D.5 Proof of Lemma 7

Let $\tau = (\mathcal{Q}, L)$ be a good transcript. Since all the answers are uniformly random in the ideal world, it is clear that one has

$$\Pr\left[\theta_{\text{id}} = \tau\right] = \frac{1}{2^{q\lambda+2n}}. \tag{15}$$

We now focus on the real world. Slightly overloading our notation, we are going to denote by F_0 the uniformly random tweakable function that is used in the real world and $\mathsf{SFMac}[F_0]$ the SFMac algorithm instantiated with this tweakable function. We are now going to lower bound the probability of observing τ in the real world. This is equivalent to lower bounding the probability that F_0 is such that $\mathsf{SFMac}[F_0]$ yields τ. This event will be denoted by $F_0 \vdash \tau$. This event is actually equivalent to the following $q+1$ equations:

$$\begin{aligned}
[F_0^{0^{t+1}}(0^n)]_{2n} &= L \\
[F_0^{0||V_1}(U_1)]_\lambda &= T_1 \\
&\vdots \\
[F_0^{0||V_q}(U_q)]_\lambda &= T_q,
\end{aligned}$$

where $U_i||V_i = [H_L(A_i, M_i)]_{n+t}$ for $i = 1, \ldots, q$. Since τ is a good transcript, we can be sure that the inputs of those conditions are actually pairwise distinct. Thus, one has

$$\Pr[\theta_{\mathrm{re}} = \tau] = \frac{1}{2^{q\lambda+2n}}, \tag{16}$$

which ends the proof of Lemma 7. □

D.6 Proof of Lemma 4

Let us denote by Θ the subset of all attainable transcripts that satisfy condition that defines bad transcripts.

Let i, j be two integers such that $1 \leq i < j \leq q$. We want to upper bound the probability that

$$I_{t,i} = I_{t,j} \text{ and } I_{n,j} = I_{n,i} \boxplus i' \boxminus j'$$

for some integers i', j' such that $0 \leq i' \leq m_i - 1$ and $0 \leq j' \leq m_j - 1$. It is clear that this probability is smaller than

$$\frac{m_i + m_j}{2^{m'}}.$$

Summing over all the pairs of messages yields

$$\Pr[\theta_{\mathrm{id}} \in \Theta_1] \leq \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} \frac{m_i + m_j}{2^{m'}} \leq \sum_{i=1}^{q-1} \frac{(q-1)m_i + \sigma}{2^{m'}} \leq \frac{2(q-1)\sigma}{2^{m'}}, \tag{17}$$

which ends the proof of Lemma 4 □

D.7 Proof of Lemma 5

Let $\mathcal{Q}$ be a good queries transcript. Since the output of the ideal world oracle is uniformly random, and by our assumption that D always queries a total of σ blocks across all its queries, one has

$$\Pr\left[\theta_{\text{id}} = \mathcal{Q}\right] = \frac{1}{2^{q\lambda+m\sigma}}.$$

Let us now focus on the real world. As in the proof of Lemma 7, we are going to denote by F_0 the uniformly random tweakable function that is used by the real world oracle. Moreover, we denote by $F_0 \vdash \mathcal{Q}$ the event where the construction FEnc, when using the tweakable function F_0, yields the queries transcript $\mathcal{Q}$. Then, one has

$$\Pr\left[\theta_{\text{re}} = \mathcal{Q}\right] = \Pr\left[F_0 \vdash \mathcal{Q}\right].$$

The event $F_0 \vdash \mathcal{Q}$ is actually equivalent to the following conditions on F_0:

$$F_0^{010||I_{t,i}}(I_{n,i} \boxplus (i'-1)) = M_i[i'] \oplus C_i[i']$$

for all $i = 1, \ldots, q$ and $i' = 1, \ldots, m_i$. Note that, since τ is a good transcript and no query can be longer than $2^n - 1$ blocks, there is no collision between the inputs of those σ conditions. Hence, it is easy to see that

$$\Pr\left[\theta_{\text{re}} = \mathcal{Q}\right] = \Pr\left[\theta_{\text{id}} = \mathcal{Q}\right],$$

which ends the proof of this lemma. □

E Details on the Security Analysis

E.1 Discussion of the bounds obtained in Section 5.1

In this appendix, we show that the rather low bounds given by the byte wise model for the second scenario in Section 5.1 can be improved by taking into account the definition of the MixColumns matrix and of the round constant difference. To see this, consider the minimum returned for two rounds, equal to 16. The byte-wise model returns a unique pattern reaching this minimum (depicted on top in Figure 8) in which the third active diagonal of the input of the second round is fully cancelled by the round constant difference. However, the difference in the third column origins from a single active byte of difference denoted δ, so is equal to $(02.\delta, \delta, \delta, 03.\delta)^T$ given the definition of MixColumns. The round constant difference is equal to $(i \oplus j, i \oplus j, i \oplus j, i \oplus j)^T$ between branch i and j, so at most two bytes of differences can be cancelled, resulting in the best characteristic being the one depicted at the bottom of Figure 8.

For 5 rounds and more, the optimal solutions returned by the model are of the form depicted in Figure 9, with in particular a succession of a fully active column originating from a single active byte that entered a MixColumns that gets cancelled by the addition of the round constant difference. As discussed previously, this sequence is impossible, and thus the bound is actually higher.

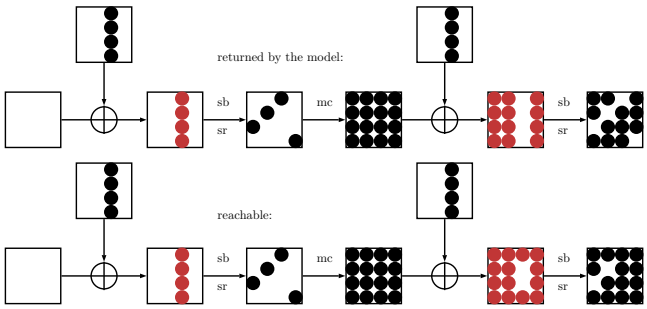

Fig. 8: Best characteristic returned by the byte-wise model (top) and actual best reachable solution (bottom).

E.2 Simple 3-round impossible differential distinguisher in the second scenario

E.3 Reminder of the best results on Deoxys-BC-256 with 128-bit key and tweak

Table 6: Some of the best results on Deoxys-BC-256 with 128-bit key and tweak.

rounds	Technique	Time	Data	Memory	ref.
8	MITM	2^{113}	2^{113}	2^{97}	[LJ19]
9	RK Imp. dif.	2^{118}	2^{118}	2^{102}	[MMS18]
10	RK boomerang	$2^{109.1}$	$2^{98.4}$	2^{88}	[ZDJ19]
	RK rectangle	$2^{114.2}$	$2^{114.2}$	$2^{112.2}$	[ZDJ19]

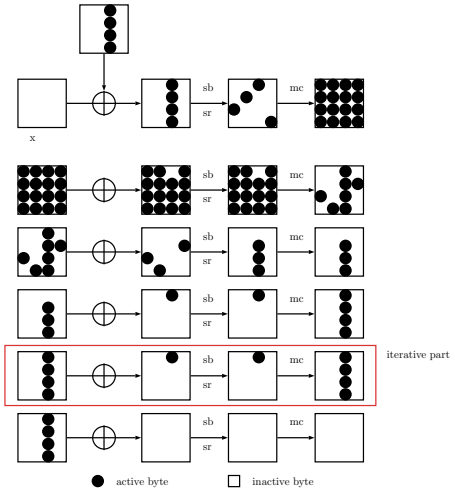

Fig. 9: For 5 rounds and more, the red iterative characteristic allows to add only 1 active Sbox per round, in the byte-wise model.

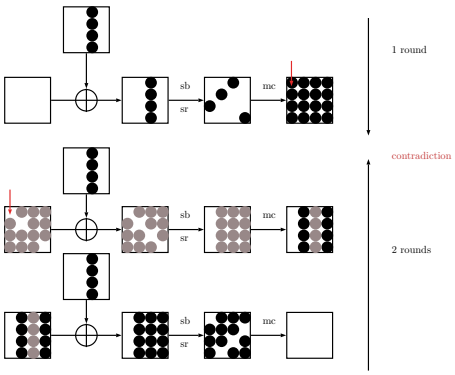

Fig. 10: 3-round impossible differential.