# Revisiting Full-PRF-Secure PMAC and Using It for Beyond-Birthday Authenticated Encryption

Eik List[1] and Mridul Nandi[2]

[1] Bauhaus-Universität Weimar, Germany
eik.list(at)uni-weimar.de
[2] Applied Statistics Unit, Indian Statistical Institute, Kolkata, India
mridul.nandi(at)gmail.com

Version of June 20, 2017

**Abstract.** This paper proposes an authenticated encryption scheme, called SIVx, that preserves BBB security also without the requirement for nonces. For this purpose, we propose a single-key BBB-secure message authentication code with $2n$-bit outputs, called PMAC2x, based on a tweakable block cipher. PMAC2x is motivated by PMAC_TBC1k by Naito; we revisit its security proof and point out an invalid assumption. As a remedy, we provide an alternative proof for our construction, and derive a corrected bound for PMAC_TBC1k.

**Keywords:** Symmetric cryptography · message authentication codes · authenticated encryption · provable security.

## 1 Introduction

***Nonce-Based Authenticated Encryption.*** Authenticated encryption (AE) schemes aim at protecting both the privacy and the integrity of submitted messages. Authenticated encryption schemes that allow to authenticate not only the encrypted message, but also associated data, are commonly known as AEAD schemes [24]. The common security notions for AE schemes concern the prevention of any leakage about the encrypted messages except for their lengths. Since stateless schemes would enable adversaries to detect a duplicate encryption of the same associated data and message under the current key, Rogaway proposed nonce-based encryption [26], where the user provides an additional nonce for every message she wants to process. In theory, the concept of nonces is simple. However, the practice has shown numerous examples of implementation failures, and settings that render it difficult to almost impossible to prevent nonce reuse (cf. [9]). Before the CAESAR competition, the majority of widely used AE schemes protected neither the confidentiality nor the integrity of messages in the case of nonce repetitions. As a consequence, a considerable number of CAESAR candidates aimed a certain level of security if nonces repeat (e.g., [1,11,12,16]).

***Parallelizable MACs in Authenticated Encryption.*** Block-cipher-based message authentication codes (MACs) are important components not only for authentication, but also as part of AE schemes, where they are used to derive an initialization vector (IV) that is then used for encryption. In particular, parallelizable MACs like PMAC [7] allow to process multiple blocks in parallel, which is beneficial for software performance on current x64 processors. Since PMAC has several further desirable properties, e. g. being online and incremental, it is not a surprise that all the CAESAR candidates cited above essentially combine a variant of PMAC (or its underlying hash function) with a block-cipher-based mode of operation for efficiently processing associated data and message.

***Beyond-Birthday-Bound AE.*** Besides performance, the quantitative security guarantees are important aspects for AE schemes. The privacy and authenticity guarantees of the AE schemes cited above are limited by the birthday bound of $O(\ell^2/2^n)$, where $n$ denotes the state size of the underlying primitive, and $\ell$ the number of blocks processed over all queries. Since the schemes above possess an $n$-bit state, a state collision that leads to attacks has significant probability after approximately $2^{n/2}$ blocks have been processed under the same key. To address this issue, Peyrin and Seurin presented Synthetic Counter in Tweak (SCT) [22], a beyond-birthday-bound (BBB) AE scheme based on a tweakable block cipher under a single key. Internally, SCT is a MAC-then-Encrypt composition: the MAC part is a PMAC-like construction, called EPWC. The encryption part is Counter-in-Tweak (CTRT), an efficient mode of operation that takes an $n$-bit nonce and an $n$-bit IV. Both EPWC and CTRT guarantee BBB security as long as nonces never repeat. However, the security of the nonce-IV-based CTRT degrades to the birthday bound with an increasing number of nonce reuses; even worse, the security of EPWC (and consequently that of SCT) immediately reduces to the birthday bound if a single nonce repeats once. In [23, p.7], the extended version of [22], the authors remarked therefore (among others) the following open problem: "[...] *to construct an AE scheme which remains BBB-secure even when nonces are arbitrarily repeated. The main difficulty is to build a deterministic, stateless, BBB-secure MAC, which is known to be notably hard*". Intuitively, an efficient block-cipher-based BBB-secure MAC with $2n$ bit output length could allow to construct such a deterministic AE scheme.[3] Thus, this work will put a large focus on the construction of BBB-secure MACs.

***Previous Work.*** Naito [19] proposed two MACs with full PRF security based on a tweakable block cipher: PMAC_TBC3K and PMAC_TBC1K. While the former requires three keys, the latter uses tweak-based domain separation to require only a single key. Extending the latter seemed a well-suited starting point for our work since such a MAC could be combined in straight-forward manner with a BBB-secure mode of encryption. Though, during our studies, we found

[3] We stress that BBB-secure AE is not new if one considers schemes with multiple primitives and keys. For the sake of space limitations, a discussion can be found in Appendix A.

**Table 1:** Previous parallel BBB-secure MACs. (T)BC = (tweakable) block cipher, $q$ = max. #queries, $m$ = max. #blocks per query, $\ell$ = max. total #blocks.

| Primitive | Construction | Keys | Output Size | Advantage | Ref. |
|---|---|---|---|---|---|
| BC | PMAC$^+$ | 3 | $n$ | $O(q^3m^3/2^{2n} + qm/2^n)$ | [29] |
| | 1K_PMAC$^+$ | 1 | $n$ | $O(qm^2/2^n + q^3m^4/2^{2n})$ | [8] |
| TBC | PMAC_TBC3K | 3 | $n$ | $O(q^2/2^{2n})$ | [19] |
| | PMAC_TBC1K | 1 | $n$ | $O(q/2^n + q^2/2^{2n})$ | [19] |
| | **PMACx** | 1 | $n$ | $O(q^2/2^{2n} + q^3/2^{3n})$ | Sec. 5 |
| | **PMAC2x** | 1 | $2n$ | $O(q^2/2^{2n} + q^3/2^{3n})$ | Sec. 4 |

that the analysis in [19] assumed internal values to be independent, which—as we will show—cannot always be guaranteed. Since the proof depended largely on this aspect, we developed an alternative analysis for our construction and derived a corrected bound for a PMAC_TBC1K-like variant with $n$-bit output. So, despite the assumption in the original proof, we confirm that Naito's MAC is secure for close to $2^{n-2}$ blocks processed under the same key.

***Contribution.*** Our contributions are threefold: first, we propose a BBB-secure parallelizable MAC, called PMAC2x, which produces $2n$-bit outputs and bases on a tweakable block cipher. Figure 1 provides a schematic illustration. As our second contribution, we briefly revisit the analysis by Naito on PMAC_TBC1K and show that we can easily adapt our proof for PMAC2x and derive a secure variant that we call PMACx which XORs both its outputs and produces only $n$-bit tags. Table 1 compares our constructions to earlier parallelizable BBB- secure MACs. As our third contribution, we combine PMAC2x with the purely IV-based variant of Counter-in-Tweak to a single-primitive, single-key deterministic authenticated encryption scheme, which we call SIVx, and which provides BBB-security without assumptions about nonces.

***Earlier Parallelizable MACs.*** A considerable amount of works considered parallel MACs; parallel XOR-MACs have already been introduced in 1995 by Bellare et al. [3]; their constructions fed the message blocks together with a counter into a primitive to obtain stateful and randomized MACs. Bernstein [6] published the Protected Counter Sum (PCS), which transformed an XOR-MAC with an independent PRF into a stateless deterministic MAC. PMAC was described by Black and Rogaway first in [7], and was slightly modified to PMAC1 in [25]. Since then, the security of PMAC has been rigorously studied in various works [13,15,18,20,21]. The first BBB-secure parallelizable MAC was proposed by Yasuda [29]; His PMAC$^+$ construction is a three-key version of PMAC which possesses two $n$-bit state values, which are processed by two independently keyed PRPs, and are XORed to produce the tag. Datta et al. [8] derived a single-key version thereof, called 1K_PMAC$^+$. While those are rate-1 designs with larger

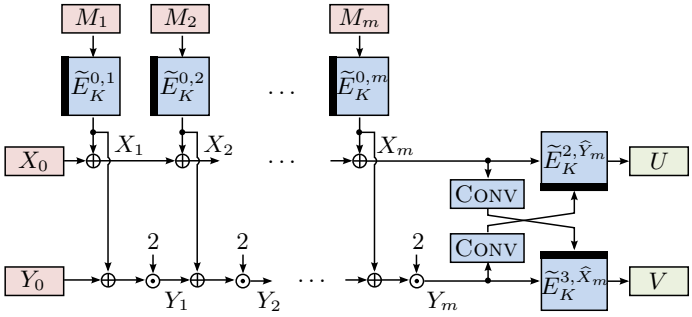


**Fig. 1:** Processing a message $M$ with PMAC2x where $M$ has $m$ blocks $(M_1, \ldots, M_m)$ after appending a $10^*$ padding to $M$. $\widetilde{E} : \{0,1\}^k \times \{0,1\}^d \times \{0,1\}^t \times \{0,1\}^n \to \{0,1\}^n$ denotes a tweakable block cipher, $\textsc{Conv} : \{0,1\}^n \to \{0,1\}^t$ a regular function, and $\odot$ multiplication in Galois-Field $\mathbb{GF}(2^n)$.

internal state, there also exist slightly less efficient proposals with smaller state. Yasuda [30] introduced PMAC with parity (PMAC/P), which processes each sequence of $r$ consecutive message blocks in PMAC-like manner, but inserts the XOR sum of those $r$ blocks as an additional block. Zhang's PMACX construction [31] generalized PMAC/P by multiplying the input with an MDS matrix before authentication. In a similar direction goes LightMAC [14], a lightweight variant similar to Bernstein's PCS. However, the security guarantees of all earlier parallelizable MACs in this paragraph are far from the optimal PRF bound.

## 2 Preliminaries

***General Notation.*** We use lowercase letters $x, y$ for indices and integers, uppercase letters $X, Y$ for binary strings and functions, calligraphic uppercase letters $\mathcal{X}, \mathcal{Y}$ for sets; $X \,\|\, Y$ for the concatenation of binary strings $X$ and $Y$, and $X \oplus Y$ for their bitwise XOR. We indicate the length of $X$ in bits by $|X|$, and write $X_i$ for the $i$-th block, $X[i]$ for the $i$-th most-significant bit of $X$, and $X[i..j]$ for the bit sequence $X[i], \ldots, X[j]$. We denote by $X \twoheadleftarrow \mathcal{X}$ that $X$ is chosen uniformly at random from the set $\mathcal{X}$. We define $\mathsf{Func}(\mathcal{X}, \mathcal{Y})$ for the set of all functions $F : \mathcal{X} \to \mathcal{Y}$, $\mathsf{Perm}(\mathcal{X})$ for the set of all permutations $\pi : \mathcal{X} \to \mathcal{X}$, and $\widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{X})$ for the set of tweaked permutations over $\mathcal{X}$ with tweak space $\mathcal{T}$. We define by $X_1, \ldots, X_j \xleftarrow{x} X$ an injective splitting of a string $X$ into blocks of $x$-bit such that $X = X_1 \,\|\, \cdots \,\|\, X_j$, $|X_i| = x$ for $1 \le i \le j-1$, and $|X_j| \le x$. For two sets $\mathcal{X}$ and $\mathcal{Y}$, let $\mathcal{X} \xleftarrow{\cup} \mathcal{Y}$ denote $\mathcal{X} \leftarrow \mathcal{X} \cup \mathcal{Y}$. A uniform random function $\rho : \mathcal{X} \to \mathcal{Y}$ is a random variable uniformly distributed over $\mathsf{Func}(\mathcal{X}, \mathcal{Y})$. Given a function $F : \mathcal{X} \to \mathcal{Y}$, we write $\mathsf{domain}(F)$ for the set of all inputs $X \in \mathcal{X}$ to $F$ that occurred *before* (i.e., excluding) the current query; similarly, we write $\mathsf{range}(F)$ for the set of all outputs $Y \in \mathcal{Y}$ that occurred before the current query. We borrow the notation for a restriction on a set from [9]: let $\mathcal{Q} \subseteq (\mathcal{X} \times \mathcal{Y} \times \mathcal{Z})^*$,

then we denote by $\mathcal{Q}_{|Y,Z} = \{(Y,Z) \mid \exists X : (X,Y,Z) \in \mathcal{Q}\}$ the restriction of $\mathcal{Q}$ to values $Y \in \mathcal{Y}$ and $Z \in \mathcal{Z}$. This generalizes in the obvious way.
For an event $E$, we denote by $\Pr[E]$ the probability of $E$. We write $\langle x \rangle_n$ for the binary representation of an integer $x$ as an $n$-bit string, or short $\langle x \rangle$ if $n$ is clear from the context, in big-endian manner, e. g., $\langle 1 \rangle_4$ would be encoded to (0001).

***Adversaries.*** An adversary $\mathbf{A}$ is an efficient Turing machine that interacts with a given set of oracles that appear as black boxes to $\mathbf{A}$. We denote by $\mathbf{A}^{\mathcal{O}}$ the output of $\mathbf{A}$ after interacting with some oracle $\mathcal{O}$. We write $\Delta_{\mathbf{A}}(\mathcal{O}^1; \mathcal{O}^2)$ for the advantage of $\mathbf{A}$ to distinguish between oracles $\mathcal{O}^1$ and $\mathcal{O}^2$. All probabilities are defined over the random coins of the oracles and those of the adversary, if any. We write $\mathbf{Adv}_F^X(q, \ell, \theta) := \max_{\mathbf{A}}\{\mathbf{Adv}_F^X(\mathbf{A})\}$ for the maximal advantage over all $X$-adversaries $\mathbf{A}$ on $F$ that run in time at most $\theta$ and pose at most $q$ queries of at most $\ell$ blocks in total to its oracles. Wlog., we assume that $\mathbf{A}$ never asks queries to which it already knows the answer.
We will provide pseudocode descriptions of the oracles, which will be referred to as games, according to the game-playing framework by Bellare and Rogaway [4]. Each game consists of a set of procedures. We define $\Pr[G(\mathbf{A}) \Rightarrow x]$ as the probability that the game $G$ outputs $x$ when given $\mathbf{A}$ as input.

**Definition 1 (TPRP Advantage).** *Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X}$ be a tweakable block cipher with non-empty key space $\mathcal{K}$ and tweak space $\mathcal{T}$. Let $\mathbf{A}$ a computationally bounded adversary with access to an oracle, where $K \twoheadleftarrow \mathcal{K}$ and $\widetilde{\pi} \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathcal{T}, \mathcal{X})$. Then, the* TPRP *advantage of $\mathbf{A}$ on $\widetilde{E}$ is defined as $\mathbf{Adv}_{\widetilde{E}}^{\mathrm{TPRP}}(\mathbf{A}) := \Delta_{\mathbf{A}}(\widetilde{E}_K; \widetilde{\pi})$.*

A MAC is a tuple of functions $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with non-empty key space $\mathcal{K}$, and a generic verification function $\textsc{Verify} : \mathcal{K} \times \mathcal{X} \times \mathcal{Y} \to \{1, \perp\}$, where for all $K \in \mathcal{K}$ and $X \in \mathcal{X}$, $\textsc{Verify}_K(X, Y)$ returns 1 iff $F_K(X) = Y$ and $\perp$ otherwise. We use $\perp$, when in place of an oracle, to always return the invalid symbol $\perp$. It is well-known that if $F$ is a secure PRF, it is also a secure MAC; however, the converse statement is not necessarily true.

**Definition 2 (PRF Advantage).** *Let $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be a function with non-empty key space $\mathcal{K}$, and $\mathbf{A}$ a computationally bounded adversary with access to an oracle, where $K \twoheadleftarrow \mathcal{K}$ and $\rho \twoheadleftarrow \mathsf{Func}(\mathcal{X}, \mathcal{Y})$. Then, the* PRF *advantage of $\mathbf{A}$ on $F$ is defined as $\mathbf{Adv}_F^{\mathrm{PRF}}(\mathbf{A}) := \Delta_{\mathbf{A}}(F_K; \rho)$.*

## 3 Definition of PMAC2x and PHASHx

This section defines the generic PMAC2x construction and its underlying hash function PHASHx. Fix integers $k, n, t, d$, with $d \geq 2$. Let $\mathcal{K} = \{0,1\}^k$ and $\mathcal{T} = \{0,1\}^t$ be non-empty sets of keys and tweaks, respectively. Moreover, derive a set of domains $\mathcal{D} := \{0,1,2,3\} = \{0,1\}^d$ which are encoded as their respective $d$-bit values, and a domain-tweak set $\mathcal{T}' := \mathcal{D} \times \mathcal{T}$. Let $\mathcal{M} \subseteq (\{0,1\}^n)^*$ denote an input space. Further, let $\widetilde{E} : \mathcal{K} \times \mathcal{T}' \times \{0,1\}^n \to \{0,1\}^n$ denote a tweakable block cipher.

**Algorithm 1** Definition of PMAC2x[$\widetilde{E}$] and its internal hash function PHASHx[$\widetilde{E}$] with a tweakable block cipher $\widetilde{E} : \mathcal{K} \times \{0,1\}^d \times \{0,1\}^t \times \{0,1\}^n \to \{0,1\}^n$. $n$, $t$, and $d$ denote state, tweak and domain sizes, respectively.

```
11: function PMAC2x[Ẽ_K](M)
12:   (X_m, Y_m) ← PHASHx[Ẽ_K](M)
13:   X̂_m ← Conv(X_m)
14:   Ŷ_m ← Conv(Y_m)
15:   U ← Ẽ_K^{2,Ŷ_m}(X_m)
16:   V ← Ẽ_K^{3,X̂_m}(Y_m)
17:   return (U ‖ V)

21: function Conv(X)
22:   if t ≥ n then
23:     return X
24:   return X[1..t]

31: function PHASHx[Ẽ_K](M)
32:   X_0 ← 0^n; Y_0 ← 0^n
33:   M* ← M ‖ 10*
34:   (M_1, ..., M_m) ←^n M*
35:   for i ← 1 to m do
36:     Z_i ← Ẽ_K^{0,⟨i⟩}(M_i)
37:     X_i ← X_{i−1} ⊕ Z_i
38:     Y_i ← 2 · (Y_{i−1} ⊕ Z_i)
39:   return (X_m, Y_m)
41: function Ẽ_K^{D,T}(X)
42:   T̃ ← ⟨D⟩_d ‖ T[1..t]
43:   return Ẽ_K^{T̃}(X)
```

We will often write $\widetilde{E}_K^{D,T}(\cdot)$ as short form of $\widetilde{E}(K, D, T, \cdot)$. $K \in \mathcal{K}$, $D \in \mathcal{D}$, and $T \in \mathcal{T}$ denote key, domain, and tweak, respectively. Conv : $\{0,1\}^n \to \{0,1\}^t$ be a regular function[4] which is used to convert the outputs of PHASHx, $X_m$ and $Y_m$, so they can be used as tweaks of $\widetilde{E}$ in the finalization step. We denote by PMAC2x[$\widetilde{E}$] and PHASHx[$\widetilde{E}$] the instantiation of PMAC2x and PHASHx with $\widetilde{E}$. Both are defined, with a default instantiation of Conv, in Algorithm 1. The message $M$ is always padded to $M^* \leftarrow M \,\|\, 10^p$ by appending first a single 1-bit and then $p$ 0-bits where $p$ is the smallest possible number of zero bits such that the length of $M^*$ is a multiple of $n$ bit.

## 4 Security Analysis of PMAC2x

**Theorem 1.** *Let $\widetilde{E}$ and* PMAC2x[$\widetilde{E}$] *be defined as in Section 3. Let $d + t = n$, and let $m < 2^t$ denote the maximum number of $n$-bit blocks of any query. Then*

$$\mathbf{Adv}^{\mathrm{PRF}}_{\mathrm{PMAC2x}[\widetilde{E}]}(q, \ell, \theta) \le \frac{2^{2d} q^2}{2 \cdot (2^n - q)^2} + \frac{2^d q^3}{3 \cdot 2^{2n}(2^n - q)} + \frac{2^d q^2}{2^n (2^n - q)} + \mathbf{Adv}^{\mathrm{TPRP}}_{\widetilde{E}}(\ell + 2q, O(\theta + \ell + 2q)).$$

The final term results from a standard argument after replacing the tweakable block cipher $\widetilde{E}$ by a random tweakable permutation $\widetilde{\pi} \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathcal{T}', \{0,1\}^n)$.
Let $\mathbf{A}$ be an adversary that makes at most $q$ queries of at most $m$ blocks each and of at most $\ell$ blocks in total. We assume, $\mathbf{A}$ does not ask duplicate queries and has the goal to distinguish between a PMAC2x[$\widetilde{\pi}$] oracle with an internally sampled tweaked permutation $\widetilde{\pi}$ and a random function $\rho : \{0,1\}^* \to \{0,1\}^{2n}$.

[4] A function is called regular iff all outputs are produced by an equal number of inputs.

**Algorithm 2** Main Game, initialization, finalization, and subroutines. Boxed statements belong exclusively to the real world.

**procedure** INITIALIZE
  $\mathsf{bad}_U \leftarrow \mathsf{false}$; $\mathsf{bad}_V \leftarrow \mathsf{false}$; $\mathcal{Q} \leftarrow \emptyset$
  $X_0 \leftarrow 0^n$; $Y_0 \leftarrow 0^n$; $b \twoheadleftarrow \{0,1\}$

**function** FINALIZE($b'$)
  $\mathsf{bad} \leftarrow \mathsf{bad}_U \vee \mathsf{bad}_V$
  **return** $b' = b \vee \mathsf{bad}$

**function** ORACLE($M$)
  $(X_m, Y_m) \leftarrow \text{PHASHx}[\widetilde{\pi}](M)$
  $\widehat{X}_m \leftarrow \text{CONV}(X_m)$
  $\widehat{Y}_m \leftarrow \text{CONV}(Y_m)$
  $U \twoheadleftarrow \{0,1\}^n$
  $V \twoheadleftarrow \{0,1\}^n$
  **if** $(\widehat{X}_m, \widehat{Y}_m) \in \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m}$ **then**
    $(U, V) \leftarrow \mathsf{Case1}(X_m, Y_m, \widehat{X}_m, \widehat{Y}_m)$
  **else if** $U \in \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge$
    $V \in \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$ **then**
    $(U, V) \leftarrow \mathsf{Case2}(X_m, Y_m, \widehat{X}_m, \widehat{Y}_m)$
  **else if** $U \in \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge$
    $V \notin \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$ **then**
    $(U, V) \leftarrow \mathsf{Case3}(\widehat{X}_m, \widehat{Y}_m, U, V)$
  **else if** $U \notin \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge$
    $V \in \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$ **then**
    $(U, V) \leftarrow \mathsf{Case4}(\widehat{X}_m, \widehat{Y}_m, U, V)$
  **else if** $U \notin \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge$
    $V \notin \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$ **then**
    $(U, V) \leftarrow \mathsf{Case5}(\widehat{X}_m, \widehat{Y}_m, U, V)$
  $\mathcal{Q} \xleftarrow{\cup} \{(\widehat{X}_m, \widehat{Y}_m, U, V)\}$
  $\widetilde{\pi}^{2,\widehat{Y}_m}[X_m] \leftarrow U$
  $\widetilde{\pi}^{3,\widehat{X}_m}[Y_m] \leftarrow V$
  **return** $(U \,\|\, V)$

**function** CASE5($\widehat{X}_m, \widehat{Y}_m, U, V$)
  **return** $(U, V)$

**function** CASE1($X_m, Y_m, \widehat{X}_m, \widehat{Y}_m$)
  **if** $X_m \in \mathsf{domain}(\widetilde{\pi}^{2,\widehat{Y}_m})$ **then**
    $\boxed{U \leftarrow \widetilde{\pi}^{2,\widehat{Y}_m}[X_m]}$
  **else**
    $\boxed{U \twoheadleftarrow \{0,1\}^n \setminus \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m})}$
  **if** $Y_m \in \mathsf{domain}(\widetilde{\pi}^{3,\widehat{X}_m})$ **then**
    $\boxed{V \leftarrow \widetilde{\pi}^{3,\widehat{X}_m}[Y_m]}$
  **else**
    $\boxed{V \twoheadleftarrow \{0,1\}^n \setminus \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})}$
  $\mathsf{bad}_U \leftarrow \mathsf{bad}_V \leftarrow \mathsf{true}$
  **return** $(U, V)$

**function** CASE2($X_m, Y_m, \widehat{X}_m, \widehat{Y}_m$)
  $\boxed{U \twoheadleftarrow \{0,1\}^n \setminus \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m})}$
  $\boxed{V \twoheadleftarrow \{0,1\}^n \setminus \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})}$
  $\mathsf{bad}_U \leftarrow \mathsf{bad}_V \leftarrow \mathsf{true}$
  **return** $(U, V)$

**function** CASE3($\widehat{X}_m, \widehat{Y}_m, U, V$)
  $\boxed{U \twoheadleftarrow \{0,1\}^n \setminus \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m})}$
  $\mathsf{bad}_U \leftarrow \mathsf{true}$
  **return** $(U, V)$

**function** CASE4($\widehat{X}_m, \widehat{Y}_m, U, V$)
  $\boxed{V \twoheadleftarrow \{0,1\}^n \setminus \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})}$
  $\mathsf{bad}_V \leftarrow \mathsf{true}$
  **return** $(U, V)$

We consider the game described in Algorithm 2. The game *without* the boxed statements coincides with a random function $\rho$, whereas the game *with* them exactly represents PMAC2x$[\widetilde{\pi}]$, performing lazy sampling for the permutations $\widetilde{\pi}^{2,\widehat{Y}_m}(\cdot)$ and $\widetilde{\pi}^{3,\widehat{X}_m}(\cdot)$, for all $\widehat{X}_m, \widehat{Y}_m \in \{0,1\}^t$. Both algorithms differ only when $\mathsf{bad}$ events occur. Hence, by the fundamental lemma of game playing [5], it holds

$$\Pr[\mathbf{A}^{\text{PMAC2x}[\widetilde{\pi}](\cdot)} \Rightarrow 1] - \Pr[\mathbf{A}^{\rho(\cdot)} \Rightarrow 1] \leq \Pr[\mathbf{A} \text{ sets } \mathsf{bad}].$$

In the remainder, we consider five cases which cover all possibilities:

– Case1: $(\widehat{X}_m, \widehat{Y}_m) \in \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m}$.
– Case2: $(\widehat{X}_m, \widehat{Y}_m) \notin \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m} \wedge U \in \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge V \in \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$.
– Case3: $(\widehat{X}_m, \widehat{Y}_m) \notin \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m} \wedge U \in \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge V \notin \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$.
– Case4: $(\widehat{X}_m, \widehat{Y}_m) \notin \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m} \wedge U \notin \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge V \in \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$.
– Case5: $(\widehat{X}_m, \widehat{Y}_m) \notin \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m} \wedge U \notin \mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m}) \wedge V \notin \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})$.

We list Case 5 only for the sake of completeness. It is easy to see that in Case 5, the output of the game is indistinguishable between the worlds. We use $M^i$, $\widehat{X}^i_m$, $\widehat{Y}^i_m$, $U^i$, $V^i$ to refer to the respective values of the $i$-th query, where $i \in [1, q]$, and assume it is the current query of the adversary. Additionally, we will use indices $j$ and $k$, where $j, k \in [1, i-1]$, to refer to previous queries.

**Case1***:* For this case, we revisit the fact that for two fixed disjoint queries $M^i$ and $M^j$, the values $X_m$ and $Y_m$ are results of two random variables. A variant of the proof is given e.g. in [19, Sect. 3.3], and revisited in the following only for the sake of completeness. Fix query indices $i \in [1, q]$ and $j \in [1, i-1]$. Let $m$ and $m'$ denote the number of blocks of the $i$-th and $j$-th query, respectively; moreover, let $X^i_m$, $Y^i_m$ denote the values $X_m$ and $Y_m$ of the $i$-th query and $X^j_{m'}$, $Y^j_{m'}$ those of the $j$-th query, respectively. $X^i_m$, $Y^i_m$, $X^j_m$, and $Y^j_m$ result from:

$$X^i_m = C^i_1 \oplus C^i_2 \oplus \cdots \oplus C^i_m \qquad Y^i_m = 2^m C^i_1 \oplus 2^{m-1} C^i_2 \oplus \cdots \oplus 2 \cdot C^i_m,$$
$$X^i_{m'} = C^j_1 \oplus C^j_2 \oplus \cdots \oplus C^j_{m'} \qquad Y^j_{m'} = 2^{m'} C^j_1 \oplus 2^{m'-1} C^j_2 \oplus \cdots \oplus 2 \cdot C^j_{m'}.$$

So, we want to bound the probability for the following equational system:

$$C^i_1 \oplus C^i_2 \oplus \cdots \oplus C^i_m = C^j_1 \oplus C^j_2 \oplus \cdots \oplus C^j_{m'}$$
$$2^m C^i_1 \oplus 2^{m-1} C^i_2 \oplus \cdots \oplus 2 \cdot C^i_m = 2^{m'} C^j_1 \oplus 2^{m'-1} C^j_2 \oplus \cdots \oplus 2 \cdot C^j_{m'}.$$

There exist three distinct subcases which cover all possibilities:

– **Subcase 1: $\mathbf{m \neq m'}$.** In this case, the equations above provide a unique solution set for two random variables; thus, the probability that the equations hold for two fixed queries is upper bounded by $1/(2^n - (i-1))^2$.
– **Subcase 2: $\mathbf{m = m'}$ and there exists $\boldsymbol{\alpha} \in [\mathbf{1}, \mathbf{m}]$ s.t. $\mathbf{C^i_\alpha \neq C^j_\alpha}$ and for all $\boldsymbol{\beta} \in [\mathbf{1}, \mathbf{m}]$ with $\boldsymbol{\beta} \neq \boldsymbol{\alpha}$ : $\mathbf{C^i_\beta = C^j_\beta}$.** In this case, both messages share only a single different output block. Thus, the equations above never hold.
– **Subcase 3: $\mathbf{m = m'}$ and there exist distinct $\boldsymbol{\beta} \in [\mathbf{1}, \mathbf{m}]$ with $\boldsymbol{\beta} \neq \boldsymbol{\alpha}$ : $C^i_{\boldsymbol{\beta}} = C^j_{\boldsymbol{\beta}}$.** In this case, the equations provide a unique solution set for two random variables, so the probability that they hold is bounded by $1/(2^n - (i-1))^2$.

So, the probability for two fixed disjoint queries $M^i$ and $M^j$ that $(X^i_m, Y^i_m) = (X^j_{m'}, Y^j_{m'})$ holds, is bounded by $1/(2^n - q)^2$. Since $\widehat{X}^i_m$ and $\widehat{Y}^i_m$ are derived from $X^i_m$ and $Y^i_m$, respectively by a regular function (and so are $\widehat{X}^j_m$ and $\widehat{Y}^j_m$ derived

from $X^j_{m'}$ and $Y^j_{m'}$, respectively), it follows that the probability of $(\widehat{X}^i_m, \widehat{Y}^i_m) = (\widehat{X}^j_{m'}, \widehat{Y}^j_{m'})$ to hold for the $i$-th and $j$-th query, with $j \in [1, i-1]$, is at most

$$(i-1) \cdot \frac{2^d}{(2^n - q)} \cdot \frac{2^d}{(2^n - q)} = \frac{2^{2d}(i-1)}{(2^n - q)^2}.$$

*Remark 1.* In its initial version, PMAC2x employed a different tweak domain $\widetilde{E}^{1,\cdot}$ if the last message block was partial, i.e., if $|M_m| < n$. This approach allowed birthday-bound collisions for the result of two messages with full $M_m$ and partial $M'_m$ since $\widetilde{E}^{0,m}$ and $\widetilde{E}^{1,m}$ defined different permutations, contradicting Subcase 2. Therefore, this revised version employs instead the approach that has also been used by Naito, i.e., to always pad the message $M^* \leftarrow (M \,\|\, 10^*)$ before using it, which ensures that Subcase 2 holds.

**Case2***:* In this case, there exists some previous query $(\widehat{X}^j_{m'}, \widehat{Y}^j_{m'}, U^j, V^j)$ s.t. $U = U^j \wedge \widehat{Y}_m = \widehat{Y}^j_{m'}$, and a distinct previous query $(\widehat{X}^k_{m''}, \widehat{Y}^k_{m''}, U^k, V^k)$ s.t. $V = V^k \wedge \widehat{X}_m = \widehat{X}^k_{m''}$. From our assumption $(\widehat{X}_m, \widehat{Y}_m) \notin \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m}$, it follows that $j \neq k$; otherwise, the current query would have stepped into Case1 instead. We can bound the probability by

$$\begin{aligned} &\Pr\left[(U = U^j \wedge \widehat{Y}_m = \widehat{Y}^j_{m'}) \wedge (V = V^k \wedge \widehat{X}_m = \widehat{X}^k_{m''})\right] \\ &\leq \Pr\left[U = U^j \wedge \widehat{Y}_m = \widehat{Y}^j_{m'} \wedge V = V^k \mid \widehat{X}_m = \widehat{X}^k_{m''}\right]. \end{aligned}$$

$U$ and $V$ are chosen independently and uniformly at random from $\{0,1\}^n$ each, and can collide with at most $i-1$ previous values $U^j$ and at most $i-1$ previous values $V^k$, respectively. For fixed $j$ and $k$, the probability for $U$ to collide with $U^j$ is upper bounded by $1/2^n$, and independently, the probability for $V$ to collide with $V^k$ is also $1/2^n$. Since the game chooses $U$ and $V$ independently from $Y_m$, the probability that $\widehat{Y}_m$ collides with $\widehat{Y}^j_{m'}$ is at most $2^d/(2^n - q)$ since we assumed that the adversary poses no duplicate queries, and therefore, $Y_m$ and $Y^j_{m'}$ are results of two random variables. Since the collision of $U = U^j$ already fixes the colliding query pair, there is no additional factor $(i-1)$ for the choice of which pairs of $\widehat{Y}_m$ and $\widehat{Y}^j_{m'}$ to collide. It follows that the probability for this case to occur at the $i$-th query is upper bounded by

$$\frac{i-1}{2^n} \cdot \frac{i-2}{2^n} \cdot \frac{2^d}{2^n - q} \leq \frac{2^d(i-1)^2}{2^{2n}(2^n - q)}.$$

**Case3***:* In this case, there exists some previous query $(\widehat{X}^j_{m'}, \widehat{Y}^j_{m'}, U^j, V^j)$ s.t. $U = U^j \wedge \widehat{Y}_m = \widehat{Y}^j_{m'}$. From our assumption $(\widehat{X}_m, \widehat{Y}_m) \notin \mathcal{Q}_{|\widehat{X}_m, \widehat{Y}_m}$ and $\widehat{Y}_m = \widehat{Y}^j_{m'}$ follows that $\widehat{X}_m \neq \widehat{X}^j_{m'}$ holds, like in Case2. We can bound the probability by

$$\begin{aligned} &\Pr\left[U = U^j \wedge \widehat{Y}_m = \widehat{Y}^j_{m'} \wedge V \notin \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})\right] \\ &\leq \Pr\left[U = U^j \wedge \widehat{Y}_m = \widehat{Y}^j_{m'} \mid V \notin \mathsf{range}(\widetilde{\pi}^{3,\widehat{X}_m})\right]. \end{aligned}$$

For a fixed $j$-th query, the probability that $\widehat{Y}_m$ collides with $\widehat{Y}^j_{m'}$ is at most $2^d/(2^n-q)$. Since $U$ is chosen uniformly at random from $\{0,1\}^n$ and independently from $Y_m$, $U$ can collide with $U^j$ with probability $1/2^n$. So, the probability of this case to occur for the $i$-th query can be upper bounded by

$$\frac{2^d}{2^n-q}\cdot\frac{i-1}{2^n}=\frac{2^d(i-1)}{2^n(2^n-q)}.$$

**Case4***:* In this case, it holds that $V=V^j\wedge\widehat{X}_m=\widehat{X}^j_{m'}$. From $(\widehat{X}_m,\widehat{Y}_m)\notin\mathcal{Q}_{|\widehat{X}_m,\widehat{Y}_m}$ and $\widehat{X}_m=\widehat{X}^j_{m'}$ follows here that $\widehat{Y}_m\neq\widehat{Y}^j_{m'}$ holds, analogously to Case2 and Case3. We can bound the probability by

$$\begin{aligned}&\Pr\left[V=V^j\wedge\widehat{X}_m=\widehat{X}^j_{m'}\wedge U\notin\mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m})\right]\\ \leq&\Pr\left[V=V^j\wedge\widehat{X}_m=\widehat{X}^j_{m'}\mid U\notin\mathsf{range}(\widetilde{\pi}^{2,\widehat{Y}_m})\right].\end{aligned}$$

Obviously, this case can be handled similarly as Case3. For a fixed $j$-th query, the probability that $\widehat{X}_m$ collides with $\widehat{X}^j_{m'}$ is at most $2^d/(2^n-q)$. Since $V$ is chosen uniformly at random from $\{0,1\}^n$ and independently from $X_m$, $V$ can collide with $V^j$ with probability $1/2^n$. So, the probability of this case to occur for the $i$-th query can also be upper bounded by

$$\frac{2^d(i-1)}{2^n(2^n-q)}.$$

Taking the terms from all cases and the union bound over at most $q$ queries gives

$$\begin{aligned}\Pr\left[\mathbf{A}\text{ sets }\mathsf{bad}\right]&\leq\sum_{i=1}^{q}\left(\frac{2^{2d}(i-1)}{(2^n-q)^2}+\frac{2^d(i-1)^2}{2^{2n}(2^n-q)}+2\cdot\frac{2^d(i-1)}{2^n(2^n-q)}\right)\\&\leq\frac{2^{2d}q^2}{2\cdot(2^n-q)^2}+\frac{2^dq^3}{3\cdot2^{2n}(2^n-q)}+\frac{2^dq^2}{2^n(2^n-q)}.\end{aligned}$$

## 5 Security Analysis of PMACx

This section considers a variant of PMAC2x, PMACx, that adds a final XOR to produce only an $n$-bit tag, following the design of PMAC_TBC1k. A schematic illustration is given in Figure 2. We revisit the assumption by Naito, and show that our proof of PMAC2x needs only a slight adaption for PMACx.

***Previous Analysis.*** Theorem 2 in [19] proves the security of PMAC_TBC1k with the help of an analysis of multi-collisions of the final chaining values ($X_m$ and $Y_m$ in our notation). Note that our notation differs from [19] to be consistent to our previous section. Define two monotone events $\mathsf{mcoll}_1$ and $\mathsf{mcoll}_2$. Let $\rho$

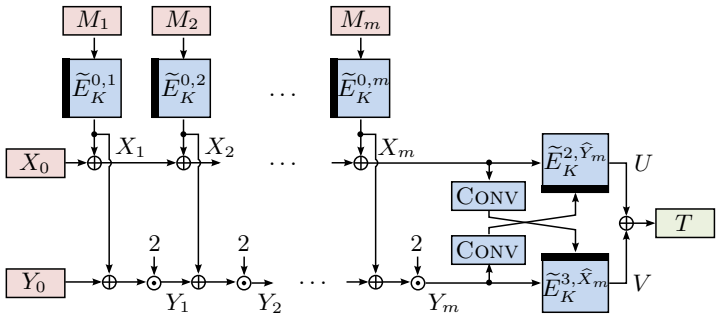


**Fig. 2:** PMACx, the variant of PMAC2x with $n$-bit output, following the design of PMAC_TBC1k.

and $\xi$ denote positive integers and define three sets $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Q}$ which store the values $\widehat{X}_m^i$, $\widehat{Y}_m^i$, and the tuples $(X_m^i, \widehat{Y}_m^i)$, respectively, of the queries $1 \le i \le q$.

$$\begin{aligned}\mathsf{mcoll}_1 &:= (\exists\, \widehat{X}_m^1, \ldots, \widehat{X}_m^\rho \in \mathcal{X} \text{ s.t. } \widehat{X}_m^1 = \ldots = \widehat{X}_m^\rho) \vee \\ &\quad\ (\exists\, \widehat{Y}_m^1, \ldots, \widehat{Y}_m^\rho \in \mathcal{Y} \text{ s.t. } \widehat{Y}_m^1 = \ldots = \widehat{Y}_m^\rho), \\ \mathsf{mcoll}_2 &:= \exists\, (X_m^1, \widehat{Y}_m^1), \ldots, (X_m^\xi, \widehat{Y}_m^\xi) \in \mathcal{Q} \text{ s.t. } (X_m^1, \widehat{Y}_m^1) = \ldots = (X_m^\xi, \widehat{Y}_m^\xi).\end{aligned}$$

The original proof further defined a monotone compound event $\mathsf{mcoll} := \mathsf{mcoll}_1 \vee \mathsf{mcoll}_2$ and used the fact that

$$\begin{aligned}\Pr\left[\mathbf{A} \text{ sets } \mathsf{bad}\right] &= \Pr\left[\mathbf{A} \text{ sets } \mathsf{bad} \wedge \mathsf{mcoll}\right] + \Pr\left[\mathbf{A} \text{ sets } \mathsf{bad} \wedge \neg\mathsf{mcoll}\right] \\ &\le \Pr\left[\mathsf{mcoll}_1\right] + \Pr\left[\mathsf{mcoll}_2\right] + \Pr\left[\mathbf{A} \text{ sets } \mathsf{bad} | \neg\mathsf{mcoll}\right].\end{aligned}$$

The analysis in [19] bounds $\Pr[\mathsf{mcoll}_1]$ as

$$\Pr[\mathsf{mcoll}_1] \le 2 \cdot 2^t \cdot \binom{q}{\rho} \cdot \left(\frac{2^{n-t}}{2^n - q}\right)^\rho \le 2^{t+1} \cdot \left(\frac{2^{n-t} \cdot eq}{\rho(2^n - q)}\right)^\rho,$$

using Stirling's approximation $x! \ge (x/e)^x$ for any $x$. Note, in PMAC_TBC1k, the domain size in PMAC2x is fixed to $d = 2$ bits. The bound above consists of the probability that $\rho$ values are all equal, $(2^{n-t}/(2^n - q))^\rho$; the fact that there are $2^t$ tweak values; and the $\binom{q}{\rho}$ possible ways to choose $\rho$ out of $q$ values. However, the bound holds *only if* the $\rho$ colliding tweaks stem from $\rho$ linearly independent random variables, which is *not* necessarily the case. Imagine a sequence of $2^m$ queries which combine pair-wise distinct blocks $\{M_i, M_i'\}$ with $M_i \ne M_i'$, for $1 \le i \le m$ position-wise, i. e., we have $2^m$ queries of $m$ blocks each: $Q^0 = (M_1, M_2, M_3, \ldots, M_m)$, $Q^1 = (M_1', M_2, M_3, \ldots, M_m)$, $Q^2 = (M_1, M_2', M_3, \ldots, M_m)$, $\ldots$, $Q^{2^m-1} = (M_1', M_2', M_3', \ldots, M_m')$. When used as queries to PMAC_TBC1k, the $2^m$ resulting values $X_m^i$, for $0 \le i \le 2^m - 1$, depend on the linear combination of only $2m$ random variables. A similar argument holds for the values $Y_m^i$, as well as for the similarly treated bound of $\mathsf{mcoll}_2$. Thus, the multi-collision bound demands a significantly more detailed analysis.

**Algorithm 3** The updated game for the security proof of PMACx. Only the double-boxed statement changes compared with the game in Algorithm 2.

21: **function** ORACLE($M$)
22: ...
42: $\widetilde{\pi}^{2,\widehat{Y}_m}[X_m] \leftarrow U$
43: $\widetilde{\pi}^{3,\widehat{X}_m}[Y_m] \leftarrow V$
44: **return** $\boxed{T \leftarrow U \oplus V}$

***Fixing the Analysis.*** From our proof for PMAC2x, we can now derive a corollary for a similar security bound for PMACx, which again can be easily transformed into a bound for PMAC_TBC1k.

**Corollary 1.** *Let $\widetilde{E}$ and* PMAC2x$[\widetilde{E}]$ *be defined as in Section 3. Let $d+t=n$, and let $m < 2^t$ denote the maximum number of $n$-bit blocks of any query. Then, it holds that* $\mathbf{Adv}^{\mathrm{PRF}}_{\mathrm{PMACx}[\widetilde{E}]}(q,\ell,\theta) \leq \mathbf{Adv}^{\mathrm{PRF}}_{\mathrm{PMAC2x}[\widetilde{E}]}(q,\ell,\theta)$.

The proof can use a game almost identical to that in Algorithm 2, where we only modify Line 44 to return the XOR of $U$ and $V$. This is shown in Algorithm 3. All further procedures and functions remain identical to those in Algorithm 2.
If $(U \,\|\, V)$ is indistinguishable from outputs of a $2n$-bit random function $\rho$, then each of the $n$-bit outputs $U$ and $V$ can be considered random. It follows, if $U$ is indistinguishable from $n$-bit values, then the XOR sum of $U \oplus V$ is also indistinguishable from a random $n$-bit value. Hence, the PRF advantage of $\mathbf{A}$ on PMACx is upper bounded by that of an adversary $\mathbf{A}'$ on PMAC2x with an equal amount of resources as $\mathbf{A}$; hence, the corollary follows.
PMACx and PMAC_TBC1k differ in two aspects: (1) PMACx defines a generic $d$-bit domain encoding and defines a conversion function Conv for deriving the inputs for the finalization; and (2) PMACx adds a final doubling for a simpler and consistent description. Clearly, none of the differences affects the distribution of final chaining values $\widehat{X}_m$ and $\widehat{Y}_m$. Hence, when fixing $d = 2$, a security result for PMACx can be easily carried over to a bound for PMAC_TBC1k.

## 6 Definition and Security Analysis of SIVx

This section defines and analyzes the deterministic AE scheme SIVx, a composition of a variant of PMAC2x with the IV-based Counter-in-Tweak mode IVCTRT. We recall the definitions of IV-based encryption and Deterministic AE security in Appendix C. Note that it is straight-forward to derive a nonce-based AE scheme by fixing the nonce length over all queries and always appending the nonce to the associated data.

**IVCTRT.** IVCTRT denotes the purely IV-based version of Counter in Tweak [23, Appendix C], which takes a $2n$-bit random IV plus the message for each

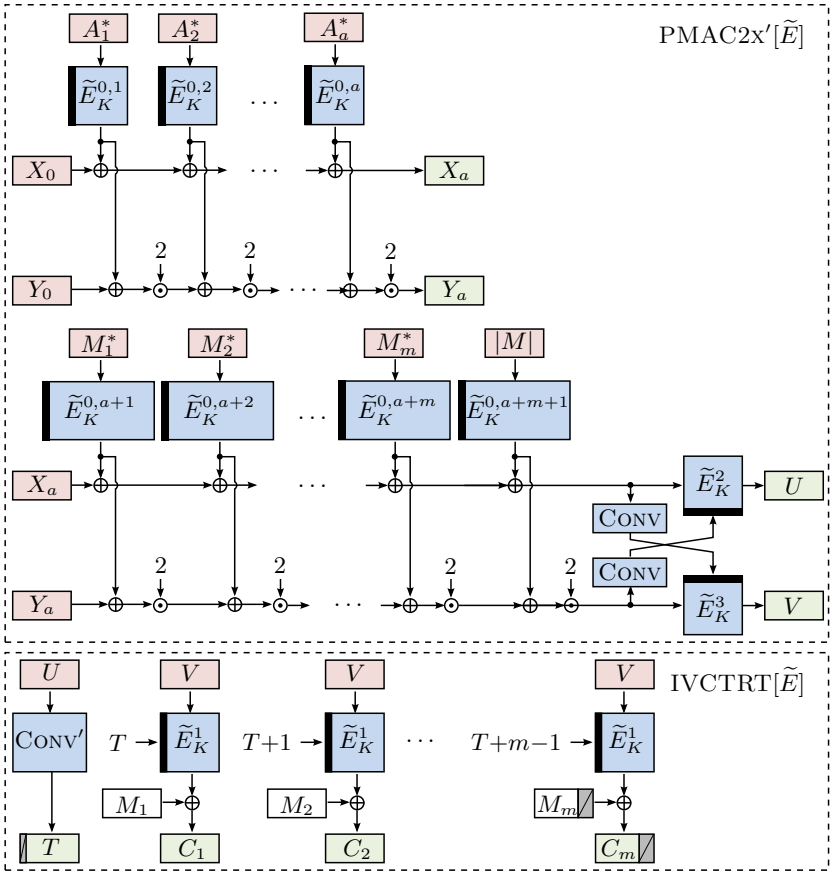


**Fig. 3:** Encryption of associated data $A$ and message $M$ with the deterministic AE scheme SIVx from the composition of PMAC2x[$\widetilde{E}$] (**top**), and the IVCTRT[$\widetilde{E}$] mode of encryption (**bottom**) [22]. $A^*$ and $M^*$ denote $A$ and $M$ after padding them with $10^*$ so that their lengths are multiples of $n$ bit each.

encryption. Let $\mathcal{T} = \{0,1\}^t$, and $\mathcal{T}' = \{0,1\} \times \mathcal{T}$. The mode first splits $(U, V) \xleftarrow{n} IV$, and uses a given tweakable block cipher $\widetilde{E} : \mathcal{K} \times \mathcal{T}' \times \{0,1\}^n \to \{0,1\}^n$ in counter mode, with $V$ as cipher input. Next, it derives $T \leftarrow \textsc{Conv}'(U)$ from $U$ with a regular function $\textsc{Conv}' : \{0,1\}^n \to \mathcal{T}$ and increments $T$ for every call to $\widetilde{E}$ using addition modulo $2^t$. We denote by IVCTRT[$\widetilde{E}$] the instantiation of IVCTRT with $\widetilde{E}$; from Theorem 1 and Appendix C in [23], we recall the following theorem:

**Theorem 2 (ivE Security of IVCTRT).** *Let $\widetilde{\pi} \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathcal{T}', \{0,1\}^n)$ be an ideal tweakable block cipher. Let* **A** *be an adversary which asks at most $q$ queries of at most $8 \le \ell \le |\mathcal{T}|$ blocks in total. Then*

$$\mathbf{Adv}^{\mathrm{IVE}}_{\mathrm{IVCTRT}[\widetilde{\pi}]}(\mathbf{A}) \le \frac{1}{2^n} + \frac{1}{|\mathcal{T}|} + \frac{4\ell \log q}{|\mathcal{T}|} + \frac{\ell \log^2(\ell)}{2^n}.$$

**Algorithm 4** Definition of SIVx$[\widetilde{E}]$ and its inverse. IVCTRT$[\widetilde{E}]$ and its inverse IVCTRT$^{-1}[\widetilde{E}]$ are identical operations. Note that PMAC2x′ does not pad the input $M^*$ since it is padded already in Encode.

```
11: function SIVx[Ẽ_K](A, M)
12:   X ← Encode(A, M)
13:   tag ← PMAC2x′[Ẽ_K](X)
14:   (U, V) ←^n tag
15:   IV ← Conv′(U) ‖ V
16:   C ← IVCTRT[Ẽ_K](IV, M)
17:   return (C, tag)

21: function IVCTRT[Ẽ_K](IV, M)
22:   (T, V) ← IV
23:   (M_1, ..., M_m) ←^n M
24:   for i ← 1 to m − 1 do
25:     C_i ← Ẽ_K^{1,T+(i−1)}(V) ⊕ M_i
26:   S_m ← Ẽ_K^{1,T+(m−1)}(V)[1..|M_m|]
27:   C_m ← S_m ⊕ M_m
28:   return C ← (C_1 ‖ ... ‖ C_m)

31: function Conv′(U)
32:   return T ← U[1..t]

41: function SIVx^{−1}[Ẽ_K](A, C, tag)
42:   (U, V) ←^n tag
43:   IV ← Conv′(U) ‖ V
44:   M ← IVCTRT^{−1}[Ẽ_K](IV, C)
45:   X ← Encode(A, M)
46:   tag′ ← PMAC2x′[Ẽ_K](X)
47:   if tag = tag′ then
48:     return M
49:   return ⊥

51: function Encode(A, M)
52:   ℓ_a ← |A| mod n
53:   ℓ_m ← |M| mod n
54:   A* ← (A ‖ 10^{n−ℓ_a−1})
55:   M* ← (M ‖ 10^{n−ℓ_m−1})
56:   L ← ⟨|M|⟩_n
57:   return (A* ‖ M* ‖ L)
```

**PMAC2x′ *and* Encode.** When processing tuples of associated data $A$ and messages $M$, we have to ensure that all distinct inputs $(A, M)$ and $(A', M')$ lead to distinct inputs to PMAC2x. For this purpose, we define an injective encoding function Encode : $\mathcal{A} \times \mathcal{M} \to \{0,1\}^*$ which is given in Algorithm 4. Encode always pads $A^* \leftarrow A \,\|\, 10^*$ and $M^* \leftarrow M \,\|\, 10^*$ so that the lengths of $A^*$ and $M^*$ are the next possible multiple of $n$ bit. It further encodes the length of $M$ in bit to an $n$-bit string named $L$, and outputs the concatenation of $M^* \leftarrow (A^* \,\|\, M^* \,\|\, L)$. Hence, every tuple $(A, M)$ results in a unique output. PMAC2x′ then processes this value $M^*$. PMAC2x′ differs from PMAC2x only in the fact that PMAC2x′ omits the padding inside PMAC2x (i.e., Line 33 in Algorithm 1 is omitted) since we already padded it before by Encode, which ensures that all distinct inputs to Encode yields distinct inputs to PMAC2x, and that the PRF proof for PMAC2x still holds.

**Lemma 1.** *Given any associated data $A, A' \in \mathcal{A}$ and messages $M, M' \in \mathcal{M}$ s.t. $(A, M) \neq (A', M')$. Then, it holds that* Encode$(A, M) \neq$ Encode$(A', M')$.

Lemma 1 is not difficult to see; a brief proof sketch is given in Appendix D.

***Definition of* SIVx.** We define the deterministic AE scheme SIVx$[\widetilde{E}]$ as the composition of PMAC2x′$[\widetilde{E}]$ and IVCTRT$[\widetilde{E}]$, as given in Algorithm 4. A schematic illustration of the encryption process is depicted in Figure 3. In general, we denote by SIVx$[F, \Pi]$ the instantiation of SIVx with a function $F$ and an IV-based encryption scheme $\Pi$ in SIVx. To use the same key in all calls to $\widetilde{E}$, we use the domains $0 = (0000)_2$ for processing as well as $2 = (0010)_2$ and

$3 = (0011)_2$ for the finalization in PMAC2x′. We encode them into the $d = 4$ most significant bits of the tweak. Inside IVCTRT, we use the single-bit domain 1 in all calls to $\widetilde{E}$ for we lose only a single bit from the IV. For concreteness, we define $X_0 = Y_0 = 0^n$.

**Theorem 3 (DAE Security of SIVx).** *Let $F : \mathcal{K}_1 \times \mathcal{A} \times \mathcal{M} \to \{0,1\}^{2n}$, and let $\Pi = (\widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ be an IV-based encryption scheme with key space $\mathcal{K}_2$ and IV space $\mathcal{IV}$. Let $K_1 \twoheadleftarrow \mathcal{K}_1$ and $K_2 \twoheadleftarrow \mathcal{K}_2$ be independent. Let $\textsc{Conv}' : \{0,1\}^n \to \{0,1\}^{n-1}$ be a regular function. Let $\mathbf{A}$ be a* DAE *adversary running in time at most $\theta$, asking at most $q$ queries of at most $8 \le \ell < 2^t$ blocks in total. Then*

$$\mathbf{Adv}^{\mathrm{DAE}}_{\mathrm{SIVx}[F,\Pi]}(\mathbf{A}) \le \mathbf{Adv}^{\mathrm{IvE}}_{\Pi}(\theta + O(\ell), q, \ell) + \mathbf{Adv}^{\mathrm{PRF}}_{F}(\theta + O(\ell), q, \ell) + \frac{q}{2^n}.$$

We defer the proof of Theorem 3 to Appendix C. Inserting the bounds from Theorems 1 and 2, we obtain the corollary below, where $F$ denotes PMAC2x′$[\widetilde{E}]$ and $\Pi$ represents IVCTRT$[\widetilde{E}]$.

**Corollary 2.** *Fix positive integers $k, n, t$ and $d = 4$. Define $d + t = n$ and let $\mathcal{T} = \{0,1\}^t$ and $\mathcal{T}' = \{0,1\}^d \times \{0,1\}^t$, and $\mathcal{IV} = \{0,1\}^{n-1}$. Let $\widetilde{E} : \mathcal{K} \times \mathcal{T}' \times \{0,1\}^n \to \{0,1\}^n$, and $\textsc{Conv}' : \{0,1\}^n \to \mathcal{IV}$ be a regular function. Let $K \twoheadleftarrow \mathcal{K}$ and $\mathbf{A}$ be a* DAE *adversary that runs in time at most $\theta$, and asks at most $q$ queries of at most $8 \le \ell < 2^t$ blocks in total. Then*

$$\mathbf{Adv}^{\mathrm{DAE}}_{\mathrm{SIVx}[\widetilde{E}]}(\mathbf{A}) \le \frac{2^{2d}q^2}{2 \cdot (2^n - q)^2} + \frac{2^d q^3}{3 \cdot 2^{2n}(2^n - q)} + \frac{2^d q^2}{2^n(2^n - q)} + \frac{4\ell \log q + 1}{2^{n-1}} + \frac{q + 1 + \ell \log^2(\ell)}{2^n} + \mathbf{Adv}^{\mathrm{TPRP}}_{\widetilde{E}}(\theta + O(2\ell + 5q), 2\ell + 5q).$$

The terms $5q$ result from the fact that we can have up to five additional calls to $\widetilde{E}$ inside PMAC2x′ because of the padding of $A$, $M$, finalization, and the additionally processed length.

## 7 Conclusion

This work revisited the PMAC_TBC1k construction by Naito for constructing a MAC with beyond-birthday-bound (BBB) security and $2n$-bit outputs, called PMAC2x. We identified a critical assumption in the previous analysis of PMAC_TBC1k and circumvented it by a new proof for PMAC2x; moreover, we could easily derive a proof for PMACx, a variant of our PMAC2x construction with $n$-bit outputs. So, we also provided a corrected bound for Naito's construction. We obtained the positive result that all three constructions provide PRF security for up to $O(q^2/2^{2n} + q^3/2^{3n})$ queries. With the help of PMAC2x, we constructed a BBB-secure AE scheme from a tweakable block cipher whose security is independent of nonces and which depends on a single primitive under a single key. We are aware that the $2n$-bit tag of SIVx requires still as many bits to be transmitted as for the $2n$-bit nonce-IV in SCT; future work could study how an appropriate truncation could reduce the transmission overhead while retaining BBB security.

## Acknowledgments

The authors would like to thank Yusuke Naito and the anonymous reviewers for their fruitful comments that helped improve our work. We address our special thanks to Kazuhiko Minematsu and Tetsu Iwata [17] who pointed out that an earlier conditional padding for partial final blocks to PMAC2x and PMACx and the combination of associated data and message in SIVx was subject to birthday-bound attacks. We revised our padding in PMAC2x and PMACx accordingly and added an injective encoding to its use in SIVx.

## Changelog

**2017-06-11** Replaced the ePrint reference of Minematsu and Iwata's paper after its acceptance in ToSC [17].

**2017-03-09** Revised the padding method in PMAC2x and PMACx to always append a $10^*$ padding to the input. Revised the definition of PMAC2x in SIVx for processing both associated data and message in SIVx, and added an injective encoding. Revised Subcase 3 in Case 1.

## References

1. Elena Andreeva, Andrey Bogdanov, Nilanjan Datta, Atul Luykx, Bart Mennink, Mridul Nandi, Elmar Tischhauser, and Kan Yasuda. COLM v1. 2016. Submission to the CAESAR competition.
2. Mihir Bellare, Anand Desai, E. Jokipii, and Phillip Rogaway. A Concrete Security Treatment of Symmetric Encryption. In *FOCS*, pages 394–403. IEEE Computer Society, 1997.
3. Mihir Bellare, Roch Guérin, and Phillip Rogaway. XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions. In Don Coppersmith, editor, *CRYPTO*, volume 963 of *LNCS*, pages 15–28. Springer, 1995.
4. Mihir Bellare and Phillip Rogaway. Code-Based Game-Playing Proofs and the Security of Triple Encryption. *IACR Cryptology ePrint Archive*, 2004:331, 2004.
5. Mihir Bellare and Phillip Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *LNCS*, pages 409–426. Springer, 2006.
6. Daniel J. Bernstein. How to Stretch Random Functions: The Security of Protected Counter Sums. *J. Cryptology*, 12(3):185–192, 1999.
7. John Black and Phillip Rogaway. A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *LNCS*, pages 384–397. Springer, 2002.
8. Nilanjan Datta, Avijit Dutta, Mridul Nandi, Goutam Paul, and Liting Zhang. Building Single-Key Beyond Birthday Bound Message Authentication Code. *IACR Cryptology ePrint Archive*, 2015:958, 2015.
9. Ewan Fleischmann, Christian Forler, and Stefan Lucks. McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In Anne Canteaut, editor, *FSE*, volume 7549 of *LNCS*, pages 196–215. Springer, 2012.

10. Christian Forler, Eik List, Stefan Lucks, and Jakob Wenzel. Efficient Beyond-Birthday-Bound-Secure Deterministic Authenticated Encryption with Minimal Stretch. In Joseph K. Liu and Ron Steinfeld, editors, *ACISP (2)*, volume 9723 of *LNCS*, pages 317–332. Springer, 2016.
11. Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust Authenticated-Encryption AEZ and the Problem That It Solves. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT (1)*, volume 9056 of *LNCS*, pages 15–44. Springer, 2015.
12. Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Deoxys v1.4. `http://competitions.cr.yp.to/caesar-submissions.html`, 2016. Third-round submission to the CAESAR competition.
13. Atul Luykx, Bart Preneel, Alan Szepieniec, and Kan Yasuda. On the Influence of Message Length in PMAC's Security Bounds. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT (1)*, volume 9665 of *LNCS*, pages 596–621. Springer, 2016.
14. Atul Luykx, Bart Preneel, Elmar Tischhauser, and Kan Yasuda. A MAC Mode for Lightweight Block Ciphers. In Thomas Peyrin, editor, *FSE*, volume 9783 of *LNCS*, pages 43–59. Springer, 2016.
15. Avradip Mandal and Mridul Nandi. An Improved Collision Probability for CBC-MAC and PMAC. *IACR Cryptology ePrint Archive*, 2007:32, 2007.
16. Kazuhiko Minematsu. Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT*, volume 8441 of *LNCS*, pages 275–292. Springer, 2014.
17. Kazuhiko Minematsu and Tetsu Iwata. Cryptanalysis of PMACx, PMAC2x, and SIVx. *IACR Transactions on Symmetric Cryptology*, 2017(2):162–176, 2017.
18. Kazuhiko Minematsu and Toshiyasu Matsushima. New Bounds for PMAC, TMAC, and XCBC. In Alex Biryukov, editor, *FSE*, volume 4593 of *LNCS*, pages 434–451. Springer, 2007.
19. Yusuke Naito. Full PRF-Secure Message Authentication Code Based on Tweakable Block Cipher. In Man Ho Au and Atsuko Miyaji, editors, *ProvSec*, volume 9451 of *LNCS*, pages 167–182. Springer, 2015.
20. Mridul Nandi. A Unified Method for Improving PRF Bounds for a Class of Blockcipher Based MACs. In Seokhie Hong and Tetsu Iwata, editors, *FSE*, volume 6147 of *LNCS*, pages 212–229. Springer, 2010.
21. Mridul Nandi and Avradip Mandal. Improved security analysis of PMAC. *J. Mathematical Cryptology*, 2(2):149–162, 2008.
22. Thomas Peyrin and Yannick Seurin. Counter-in-Tweak: Authenticated Encryption Modes for Tweakable Block Ciphers. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO I*, volume 9814 of *LNCS*, pages 33–63. Springer, 2016.
23. Thomas Peyrin and Yannick Seurin. Counter-in-Tweak: Authenticated Encryption Modes for Tweakable Block Ciphers. *IACR Cryptology ePrint Archive*, 2015:1049, Version from May 27 2016.
24. Phillip Rogaway. Authenticated-Encryption with Associated-Data. In *ACM Conference on Computer and Communications Security*, pages 98–107, 2002.
25. Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In Pil Joong Lee, editor, *ASIACRYPT*, volume 3329 of *LNCS*, pages 16–31. Springer, 2004.
26. Phillip Rogaway. Nonce-Based Symmetric Encryption. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *LNCS*, pages 348–359. Springer, 2004.

27. Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 373–390. Springer, 2006.
28. Thomas Shrimpton and R. Seth Terashima. A Modular Framework for Building Variable-Input-Length Tweakable Ciphers. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT (1)*, volume 8269 of *Lecture Notes in Computer Science*, pages 405–423. Springer, 2013.
29. Kan Yasuda. A New Variant of PMAC: Beyond the Birthday Bound. In Phillip Rogaway, editor, *CRYPTO*, volume 6841 of *LNCS*, pages 596–609. Springer, 2011.
30. Kan Yasuda. PMAC with Parity: Minimizing the Query-Length Influence. In Orr Dunkelman, editor, *CT-RSA 2012*, pages 203–214. Springer, 2012.
31. Yusi Zhang. Using an Error-Correction Code for Fast, Beyond-Birthday-Bound Authentication. In Kaisa Nyberg, editor, *CT-RSA*, pages 291–307. Springer, 2015.

## Supporting Material

## A Previous BBB-Secure Authenticated Encryption

BBB-secure AE has received attention earlier; at ASIACRYPT'13, Shrimpton and Terashima [28] proposed Protected IV (PIV), a three-pass variable-input-length cipher which can be transformed into a robust AE scheme. Shrimpton and Terashima proposed two variants of PIV, one of which provided beyond-birthday-bound security. Though, it requires one more pass over the message than our proposal, and is composed of a universal hash function to extend the tweak size of the underlying primitive, as has already been stressed by Peyrin and Seurin. Thus, PIV requires rather large keys, plus it is neither single-key nor single-primitive. Furthermore, Forler et al. [10] proposed Deterministic Counter-in-Tweak (DCT), which can be seen as a two-pass variant of Protected IV or the Hash-Counter-Hash design in general. DCT also employs Counter-in-Tweak for BBB security. Though, that construction is also a composition of a universal hash function to extend the tweak size, plus an encryption scheme, plus an additional call to a $2n$-bit permutation. Thus, it is again not single-primitive and requires a hash function to extend the tweak size. In contrast to both PIV and DCT, SCT as well as our proposal SIVx combine all three properties: single-key, a fixed-tweak-length single-primitive, and BBB security.
We would like to stress that a significant number of sponge- or—in general — permutation-based AE schemes exist which also provide high security. Last but not least, several heuristics have evolved very recently. However, we see BBB-secure block-cipher-based AE schemes as well as hash functions and MACs as their components not in direct competition to wide-state designs, but rather as a complementary line of research. Moreover, BBB-secure designs can also be instantiated with wide-block primitives to provide very high levels of security.

## B Notions for IV-based Encryption and Deterministic Authenticated Encryption

We define $\$^{\mathcal{O}}$ for an oracle that, given an input $X$, chooses uniformly at random a value $Y$ equal in length of the expected output, $|Y| = |\mathcal{O}(X)|$, and returns $Y$. We assume that $\$^{\mathcal{O}}$ performs lazy sampling, i.e., $\$^{\mathcal{O}}(X)$ returns the same value $Y$ when queried with the same input $X$. We often omit the key for brevity, e.g., $\$^{\widetilde{\mathcal{E}}}(X)$ will be short for $\$^{\widetilde{\mathcal{E}}_K}(X)$. If two oracles $\mathcal{O}_i$, $\mathcal{O}_j$ represent a family of algorithms indexed by inputs, the indices must match: e. g., when $\widetilde{\mathcal{E}}_K(A, M)$ and $\widetilde{\mathcal{D}}_K(A, C, T)$ represent encryption and decryption algorithms with a fixed key $K$ and indexed by $A$, then $\widetilde{\mathcal{E}}_K \hookrightarrow \widetilde{\mathcal{D}}_K$ says that $\mathbf{A}$ first queries $\widetilde{\mathcal{E}}_K(A, M)$ and later $\widetilde{\mathcal{D}}_K(A, \widetilde{\mathcal{E}}(A, M))$.
An IV-based encryption scheme [2] is a tuple $\mathit{\Pi} = (\mathcal{E}, \mathcal{D})$ of encryption and decryption algorithms $\mathcal{E} : \mathcal{K} \times \mathcal{IV} \times \mathcal{M} \to \mathcal{C}$ and $\mathcal{D} : \mathcal{K} \times \mathcal{IV} \times \mathcal{C} \to \mathcal{M}$, with associated non-empty key space $\mathcal{K}$, non-empty IV space $\mathcal{IV}$, and where $\mathcal{M}, \mathcal{C} \subseteq \{0,1\}^*$ denote message and ciphertext space, respectively.

**Definition 3 (ivE Advantage).** *Let $\mathit{\Pi} = (\mathcal{E}, \mathcal{D})$ be an IV-based encryption scheme with associated key space $\mathcal{K}$ and IV space $\mathcal{IV}$. Let $K \twoheadleftarrow \mathcal{K}$, and let $\mathbf{A}$ be a computationally bounded adversary with access to an oracle $\mathcal{O}$. On any input $M \in \mathcal{M}$, the game chooses $IV \twoheadleftarrow \mathcal{IV}$ and returns $(IV, \mathcal{O}(IV, M))$. The* ivE *advantage of $\mathbf{A}$ with respect to $\mathit{\Pi}$ is defined as $\mathbf{Adv}^{\mathrm{ivE}}_{\mathit{\Pi}}(\mathbf{A}) := \Delta_{\mathbf{A}}(\mathcal{E}_K; \$^{\mathcal{E}})$.*

A deterministic AE scheme (with associated data) is a tuple $\widetilde{\mathit{\Pi}} = (\widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ of deterministic algorithms $\widetilde{\mathcal{E}} : \mathcal{K} \times \mathcal{A} \times \mathcal{M} \to \mathcal{C} \cup \mathcal{T}$ and $\widetilde{\mathcal{D}} : \mathcal{K} \times \mathcal{A} \times \mathcal{C} \times \mathcal{T} \to \mathcal{M} \cup \{\perp\}$ with non-empty key space $\mathcal{K}$, associated-data space $\mathcal{A}$, message space $\mathcal{M}$, ciphertext space $\mathcal{C}$, and tag space $\mathcal{T}$ [27]. We restrict to $\mathcal{A}, \mathcal{M}, \mathcal{C}, \mathcal{T} \subseteq \{0,1\}^*$. For each $K \in \mathcal{K}$, $A \in \mathcal{A}$, and $M \in \mathcal{M}$, the output $(C, T) \leftarrow \widetilde{\mathcal{E}}_K(A, M)$ is such that $|C| = |M|$ and $|T| = \tau$ for fixed stretch $\tau$. $\widetilde{\mathcal{D}}_K(A, C, T)$ outputs the corresponding message $M$ iff $(A, C, T)$ is valid, and $\perp$ otherwise. We assume correctness, i.e., for all $K, A, M \in \mathcal{K} \times \mathcal{A} \times \mathcal{M}$, it holds that $\widetilde{\mathcal{D}}_K(A, \widetilde{\mathcal{E}}_K(A, M)) = M$. Moreover, we assume tidiness, i.e., if there exists an $M$ such that $\widetilde{\mathcal{D}}_K(A, C, T) = M$, then it holds that $\widetilde{\mathcal{E}}_K(A, \widetilde{\mathcal{D}}_K(A, C, T)) = (C, T)$.

**Definition 4 (DAE Advantage [27]).** *Let $\widetilde{\mathit{\Pi}} = (\widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ be a DAE scheme with associated key, input, and output spaces as above. Let $K \twoheadleftarrow \mathcal{K}$ and let $\mathbf{A}$ denote a computationally bounded adversary with access to two oracles $\mathcal{O}_1$ and $\mathcal{O}_2$ each such that $\mathbf{A}$ never queries $\mathcal{O}_1 \hookrightarrow \mathcal{O}_2$. Then, the* DAE *advantages of $\mathbf{A}$ with respect to $\widetilde{\mathit{\Pi}}$ is defined as $\mathbf{Adv}^{\mathrm{DAE}}_{\widetilde{\mathit{\Pi}}}(\mathbf{A}) := \Delta_{\mathbf{A}}(\widetilde{\mathcal{E}}_K, \widetilde{\mathcal{D}}_K; \$^{\widetilde{\mathcal{E}}}, \perp)$.*

## C Security Proof of SIVx

This section gives the proof of Theorem 3.

*Proof.* The proof is analogous to the security proof of NSIV (the generalization of SCT) in [23]. As an initial step, we replace $\widetilde{E}$ by an ideal tweaked permutation $\widetilde{\pi} \twoheadleftarrow \widetilde{\mathsf{Perm}}(\mathcal{T}', \{0,1\}^n)$. Clearly, the maximal advantage for an adversary to distinguish between both settings is bounded by the last term of our theorem. In the remainder, let $\widetilde{\Pi} = \text{SIVX}[F, \Pi]$, and denote by $\widetilde{\mathcal{E}}[F_{K_1}, \Pi_{K_2}]$ and $\widetilde{\mathcal{D}}[F_{K_1}, \Pi_{K_2}^{-1}]$ the encryption and decryption algorithms of $\widetilde{\Pi}$ instantiated with $F$ and $\Pi$ under independent keys $K_1$ and $K_2$, respectively. We will describe a game-based proof over a sequence of three games. We denote by

$$\mathcal{G}_1 := (\widetilde{\mathcal{E}}[F_{K_1}, \Pi_{K_2}], \widetilde{\mathcal{D}}[F_{K_1}, \Pi_{K_2}^{-1}]) \text{ and}$$
$$\mathcal{G}_4 := (\$^{\widetilde{\mathcal{E}}}, \bot)$$

the two worlds that $\mathbf{A}$ must distinguish between, where $\$^{\widetilde{\mathcal{E}}}$ returns on input $(A, M)$ a tuple of random strings of length $2n$ and $|\widetilde{\mathcal{E}}[F_{K_1}, \Pi_{K_2}](A, M)|$, respectively. We further use two intermediate worlds

$$\mathcal{G}_2 := (\widetilde{\mathcal{E}}[\rho, \Pi_{K_2}], \widetilde{\mathcal{D}}[\rho, \Pi_{K_2}^{-1}]) \text{ and}$$
$$\mathcal{G}_3 := (\widetilde{\mathcal{E}}[\rho, \Pi_{K_2}], \bot).$$

In $\mathcal{G}_2$ and $\mathcal{G}_3$, we replace $F_{K_1}$ by a random function $\rho \twoheadleftarrow \mathsf{Func}(\mathcal{A} \times \mathcal{M}, \{0,1\}^{2n})$, in both encryption and decryption algorithms. In $\mathcal{G}_3$, we also replace the real decryption algorithm $\widetilde{\mathcal{D}}$ by the $\bot$ oracle. We denote by

$$\Delta_{i,j} := \left|\Pr\left[\mathbf{A}^{\mathcal{G}_i} \Rightarrow 1\right] - \Pr\left[\mathbf{A}^{\mathcal{G}_j} \Rightarrow 1\right]\right|$$

the advantage of $\mathbf{A}$ in distinguishing world $\mathcal{G}_i$ from $\mathcal{G}_j$. First, we upper bound

$$\Delta_{3,4} := \left|\Pr\left[\mathbf{A}^{\mathcal{G}_3} \Rightarrow 1\right] - \Pr\left[\mathbf{A}^{\mathcal{G}_4} \Rightarrow 1\right]\right|$$

Consider the following adversary $\mathbf{A}'$ against the IVE security of $\Pi$: Let $\mathcal{O} \in \{\widetilde{\mathcal{E}}[\rho, \Pi_{K_2}], \$^{\widetilde{\mathcal{E}}}\}$ be the first oracle that $\mathbf{A}'$ has access to. $\mathbf{A}'$ runs $\mathbf{A}$. When $\mathbf{A}$ asks an encryption query $(A, M)$, $\mathbf{A}'$ queries $\mathcal{O}(M)$ and obtains an answer $(IV, C)$. It derives $(T, V) \leftarrow IV$, samples uniformly at random a value $U$ in the set of preimages $\text{CONV}^{-1}(T)$, derives $\text{TAG} \leftarrow (U \,\|\, V)$, and returns $(C, \text{TAG})$ to $\mathbf{A}$. When $\mathbf{A}$ asks a decryption query, $\mathbf{A}'$ simply returns $\bot$. When $\mathbf{A}$ halts and outputs a bit, $\mathbf{A}'$ simply outputs the same bit. Independently from whether $\mathcal{O}$ is the real or random encryption, the function $\rho$ outputs uniformly random values $(T \,\|\, V)$. Thus, $U$ is also uniformly random since CONV is a regular function. So, $\mathbf{A}'$ perfectly simulates $\mathcal{G}_3$ if $\mathcal{O}$ is $\widetilde{\mathcal{E}}[\rho, \Pi_{K_2}]$, and perfectly simulates $\mathcal{G}_4$ if $\mathcal{O}$ is $\$^{\widetilde{\mathcal{E}}}$. Clearly, $\mathbf{A}$ and $\mathbf{A}'$ differ neither in the number of asked queries, total blocks nor maximal query length, but only in the used time that $\mathbf{A}'$ needs to simulate $\mathbf{A}$. Hence, it holds that

$$\Delta_{3,4} \leq \mathbf{Adv}_{\Pi}^{\text{IVE}}(\mathbf{A}').$$

Next, we upper bound

$$\Delta_{1,2} := \left|\Pr\left[\mathbf{A}^{\mathcal{G}_1} \Rightarrow 1\right] - \Pr\left[\mathbf{A}^{\mathcal{G}_2} \Rightarrow 1\right]\right|$$

Let $\mathbf{A}''$ be a PRF adversary on $F$. Denote by $\mathcal{O} \in \{F_{K_1}, \rho\}$ the oracle that $\mathbf{A}''$ has access to. $\mathbf{A}''$ chooses a random $K_2 \twoheadleftarrow \mathcal{K}_2$ and runs $\mathbf{A}$. When $\mathbf{A}$ makes an encryption query $(A, M)$ or a decryption query $(A, C, \text{TAG})$, $\mathbf{A}''$ simply executes SIVx where it replaces all calls to $F_{K_1}(\cdot, \cdot)$ by a call to its oracle $\mathcal{O}$. When $\mathbf{A}$ halts and outputs a bit, $\mathbf{A}''$ outputs the same bit. Again, $\mathbf{A}''$ perfectly simulates $\mathcal{G}_1$ if $\mathcal{O}$ is $F_{K_1}$ for random $K_1$, and also perfectly simulates $\mathcal{G}_2$ if $\mathcal{O}$ is $\rho$. Clearly, $\mathbf{A}$ and $\mathbf{A}'$ differ neither in the number of queries, total blocks nor maximal query length, but only in the used time that $\mathbf{A}'$ needs to simulate $\mathbf{A}$. So, it holds that

$$\Delta_{1,2} \leq \mathbf{Adv}_F^{\text{PRF}}(\mathbf{A}'').$$

It remains to upper bound

$$\Delta_{2,3} := \left|\Pr\left[\mathbf{A}^{\mathcal{G}_2} \Rightarrow 1\right] - \Pr\left[\mathbf{A}^{\mathcal{G}_3} \Rightarrow 1\right]\right|$$

Consider $\mathcal{G}_2$ as the ideal world and $\mathcal{G}_3$ as the real world, and define as *bad* transcript a transcript that contains a valid decryption query, i. e., the decryption oracle returned some $M \neq \perp$ for some decryption query of $\mathbf{A}$. Otherwise, a transcript $\tau$ is *good* if all its contained decryption queries were answered by $\perp$. We consider two cases here:

- The transcript of $\mathbf{A}$'s queries contains a valid decryption query.
- The transcript contains *no* valid decryption query.

The second case is easy to analyze: Since the encryption oracles are equal in both worlds, and since the decryption oracle always outputs $\perp$ in $\mathcal{G}_3$, it follows for any good transcript $\tau$ that the advantage of $\mathbf{A}$ to distinguish between both worlds is zero.

It remains to upper bound the probability that a transcript $\tau$ from $\mathcal{G}_2$ is *bad*, i. e., contains a valid decryption query $(A, C, \text{TAG})$. For such a valid decryption query, we denote $\text{TAG} := (U \,\|\, V)$, $T := \text{CONV}(U)$, $IV := (T \,\|\, V)$, and $M := \widetilde{\mathcal{D}}[\rho, \Pi_{K_2}^{-1}](IV, C)$. First, assume that there was a previous encryption query $(A, M)$ that returned $(C, \text{TAG}')$. Since we assumed that the adversary never asks a decryption query $(A, C, \text{TAG}')$ if a previous encryption query $(A, M)$ returned $(C, \text{TAG}')$, it necessarily holds that $\text{TAG} \neq \text{TAG}' = \rho(A, M)$. Thus, the decryption oracle necessarily returns $\perp$ then. So, the adversary cannot profit from results of old queries.

In the opposite case, assume there was no previous encryption query $(A, M)$. Then, the output of $\rho(A, M)$ is sampled at random from $\{0,1\}^{2n}$ during the decryption of some $(A, C, \text{TAG})$. In this case, $\mathbf{A}$ has to guess the random output of $\rho(A, M)$. Since $\rho$ is a uniform random function, the success probability of $\mathbf{A}$ is at most $q/2^n$ over $q$ queries. From the fundamental Theorem of game-playing follows that the advantage of $\mathbf{A}$ to distinguish both worlds is given by the probability that a transcript is bad, which is upper bounded by

$$\Delta_{2,3} \leq \frac{q}{2^n}.$$

The result in Theorem 3 follows from applying the triangle inequality:

$$\mathbf{Adv}_{\widetilde{\Pi}}^{\text{DAE}}(\mathbf{A}) \leq \Delta_{1,2} + \Delta_{2,3} + \Delta_{3,4}.$$

## D Injectivity of Encode

*Proof of Lemma 1.*. The proof is easy to see and therefore only brief. First, we recall a well-known property of the $10^*$-padding: given any two bit-strings $X, X' \in \{0,1\}^*$ of equal length $|X| = |X'|$. Then, if they are both padded so that their lengths after padding are the next multiple of some non-negative integer $n$, i.e., $X^* \leftarrow X \,\|\, 10^*$ and $X'^* \leftarrow X' \,\|\, 10^*$, then $(X^* = X'^*) \iff (X = X')$.
Now, we can define non-negative integers $a$, $m$, $a'$, $m'$ for the numbers of blocks in $A$, $M$, $A'$, and $M'$, respectively: $A = (A_1, \ldots, A_a)$, $M = (M_1, \ldots, M_m)$, $A' = (A'_1, \ldots, A'_{a'})$ and $M' = (M'_1, \ldots, M'_{m'})$. For $\textsc{Encode}(A, M) = \textsc{Encode}(A', M')$ to hold, it must hold that $(A^* \,\|\, M^*) = (A'^* \,\|\, M'^*)$. We distinguish between two mutually exclusive cases that cover all possibilities:
**Case 1:** $|A| = |A'|$**.** In this case, the $10^*$ padding will ensure that $|A^*| = |A'^*|$. To have $(A^* \,\|\, M^*) = (A'^* \,\|\, M'^*)$, it must hold that $A^* = A'^*$, which again implies $A = A'$ from our property at the begin of this proof. Since we defined that $(A, M) \neq (A', M')$, this implies further that $M \neq M'$. We have two subcases: (1) $|M| \neq |M'|$ and (2) $|M| = |M'|$. We can instantly exclude Subcase (1) since in this case, the final block $L \neq L'$, and therefore, $\textsc{Encode}(A, M)$ never equals $\textsc{Encode}(A', M')$. In Subcase (2), the $10^*$ padding ensures that $M^* = M'^*$ holds iff $M = M'$. This leads to $(A, M) = (A', M')$, which is a contradiction to our assumption that we want a collision for distinct $(A, M) \neq (A', M')$.
**Case 2:** $|A| \neq |A'|$**.** Since $A$ and $A'$ are always padded with $10^*$ such that their respective lengths are the next possible multiples of $n$, $(A^* \,\|\, M^*) = (A'^* \,\|\, M'^*)$ can only hold if $|M^*| \neq |M'^*|$. Since $|M^*|$ and $|M'^*|$ are multiples of $n$ bit, it follows for the lengths of the original messages that $|M| \neq |M'|$. Though, this implies that the length blocks differ: $L \neq L'$. Therefore, it cannot hold $\textsc{Encode}(A, M) = \textsc{Encode}(A', M')$. Our claim that $\textsc{Encode}$ is injective follows.