Research Article

Bishwajit Chakraborty* and Mridul Nandi

The mF mode of authenticated encryption with associated data

<https://doi.org/10.1515/jmc-2020-0054>

received March 28, 2020; accepted December 22, 2021

Abstract: In recent years, the demand for lightweight cryptographic protocols has grown immensely. To fulfill this necessity, the National Institute of Standards and Technology (NIST) has initiated a standardization process for lightweight cryptographic encryption. NIST's call for proposal demands that the scheme should have one *primary* member that has a key length of 128 bits, and it should be secure up to $2^{50} - 1$ byte queries and 2^{12} computations. In this article, we propose a tweakable block cipher (TBC)-based authenticated encryption with associated data (AEAD) scheme, which we call mF. We provide authenticated encryption security analysis for mF under some weaker security assumptions (stated in the article) on the underlying TBC. We instantiate a TBC using block cipher and show that the TBC achieves these weaker securities, provided the key update function has high periodicity. mixFeed is a round 2 candidate in the aforementioned lightweight cryptographic standardization competition. When we replace the key update function with the key scheduling function of Advanced Encryption Standard (AES), the mF mode reduces to mixFeed. Recently, the low periodicity of AES key schedule is shown. Exploiting this feature, a practical attack on mixFeed is reported. We have shown that multiplication by primitive element satisfies the high periodicity property, and we have a secure instantiation of mF, a secure variant of mixFeed.

Keywords: mixFeed, multi-commitment prediction, μ -TPRP, TBC, lightweight, BC

MSC 2020: 68p25, 94a60, 94a62

1 Introduction

Lightweight cryptography. Lightweight cryptography is concerned with a huge variety of resource-limited devices including Internet of Things end nodes and radio-frequency identification tags [1]. For a resource-constrained environment, the standard cryptographic protocols such as Advanced Encryption Standard (AES) [2] and SHA3 [3], which work well together, within computer systems, are difficult to implement because of implementation size, speed or throughput, and energy consumption. The lightweight cryptography trades off implementation cost, speed, security, performance, and energy consumption on resource-limited devices. The purpose of lightweight cryptography is to consume less memory, less computing resource, and less power supply, and to provide a weaker but satisfactory security solution that can work in resource-limited devices. As a consequence, lightweight cryptographic protocols are expected to be simpler and faster as compared to conventional cryptographic protocols.

In the last decade, the demand for lightweight cryptographic protocols has grown immensely. To fulfill this necessity, the National Institute of Standards and Technology (NIST) has initiated a standardization process for lightweight cryptographic (LWC) encryption schemes. NIST's call for proposal [4] demands that a scheme should have one *primary* member that has a key length of 128 bits, and it should be secure up to

* Corresponding author: Bishwajit Chakraborty, Applied Statistics Unit, Indian Statistical Institute, Kolkata, India, e-mail: bishu.math.ynwa@gmail.com

Mridul Nandi: Applied Statistics Unit, Indian Statistical Institute, Kolkata, India, e-mail: mridul.nandi@gmail.com

$2^{50} - 1$ byte queries and 2^{112} computations. Due to the wide availability of well-secured (tweakable) block ciphers, they are a popular choice as a building block while designing lightweight cryptographic protocols. This popularity is evident from the fact that 25 out of 56 submissions to round 1 of LWC competition were based on (tweakable) block ciphers, and subsequently, 15 were promoted to round 2.

(Tweakable) block cipher. A block cipher is a cryptographic primitive consisting of two algorithms, namely, (E, D) such that the deterministic function E takes a fixed-length key and a fixed-length (also called a block) message as input to produce a single block of ciphertext output. In notation, $E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ is a function such that, for all $K \in \{0, 1\}^k$, E_K is a permutation. For any given $K \in \{0, 1\}^k$ and any $M, C \in \{0, 1\}^n$, we have $D(K, C) = M$ if and only if $E(K, M) = C$.

A tweakable block cipher (TBC) [5] is a deterministic function $\tilde{E}$, which takes a fixed-length tweak along with a fixed-length key and a block of message to output a single block of ciphertext, such that it acts as a family of block ciphers. In notation, $\tilde{E} : \{0, 1\}^t \times \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ is a function such that there exists a family of block ciphers $\{E_{tw}\}$ such that for, each $tw \in \{0, 1\}^t$, $\tilde{E}(tw, K, M) = E_{tw}(K, M)$.

TBC-based AEAD. For many AEAD modes [6–10] with an underlying TBC $\tilde{E}$ and a given key K , the encryption algorithm works as follows. It takes an initial input IV and then computes $Y_0 = \tilde{E}(tw_0, K, IV)$, where the tweak tw_0 depends on the nonce and perhaps, some other parameters. To process the i th block of message (or associated data), it uses a feedback function FB , which takes Y_{i-1} and M_i as inputs and X_i and C_i as outputs (only X_i in case of AD). It then outputs $Y_i = \tilde{E}(tw_0, K, X_i)$, which is used to process the next block of associated data or message. After processing the entire message, the next TBC output is used as the input for the tag generation protocol. For such an AEAD scheme with an appropriate feedback function FB , the privacy and the forgery advantage of any adversary can be bounded as follows:

$$\mathbf{Adv}_{\text{AEAD}}^{\text{priv}}(T, D) \leq \mathbf{Adv}_{\tilde{E}}^{\text{tprp}}(T, D), \quad (1.1)$$

$$\mathbf{Adv}_{\text{AEAD}}^{\text{forge}}(T, D) \leq \mathbf{Adv}_{\tilde{E}}^{\text{tprp}}(T, D) + O\left(\frac{D}{2^n}\right), \quad (1.2)$$

where T and D are respectively, the time and data complexity of the adversaries playing the security games and n is the TBC state size. Hence, the security of such a TBC-based AEAD scheme can be bounded by bounding the TPRP advantage of the underlying TBC.

Security of some instantiations of TBC based on block ciphers. A dedicated TBC construction is conjectured to have security, and hence, we can instantiate the mode by such a secure TBC. In addition to the dedicated constructions, there are some known constructions of TBC based on a block cipher. For example, XEX [11] is shown to have birthday-bound security.

AEAD modes such as Remus-N [7] and mixFeed [6] use explicit block cipher-based TBC constructions as their underlying TBC. Given a block cipher E and a feedback function ρ , the ICE1 TBC (used in Remus-N1) uses a key derivation function KDF, which takes inputs K and N and outputs $L = \rho(E_K(N))$. Given input X and tweak (N, ω, i) , it then outputs $E_{K'}(X)$, where $K' = 2^i L \oplus \omega$. As described in ref. [12], an adversary can make D queries with a fixed input 0^n and varying the tweaks to get D outputs $Y_1, \dots, Y_D$. Notice that ICE1 uses two block cipher calls, one is used as the key derivation function and the second one is used as the encryption function with the key that is derived from the key derivation output of the first. So, let $K_1, \dots, K_D \in \{0, 1\}^k$ be those intermediate keys. Now, if the adversary pre-computes $Y'_1, \dots, Y'_T$ by making T primitive block cipher calls with input 0^n and keys $K'_1, \dots, K'_T$, then the probability that $Y_i = Y'_j$, and hence, $K_i = K'_j$ for some $1 < i < D, 1 < j < T$ is bounded by $\frac{DT}{2^k}$. Hence,

$$\mathbf{Adv}_{\text{ICE1}}^{\text{tprp}}(T, D) \approx \frac{DT}{2^k}.$$

Note that while constructing any TBC, it is generally desired that it should achieve security with a T close to 2^k . For instance, according to NIST [4], $T \geq 2^{112}$ for $n \geq 128$ and $T \geq 2^{224}$ when $n = 256$. Hence, a bound such as above doesn't provide satisfactory security as it limits the size of D . Accordingly, given $D \geq 2^{50}$, ICE1 doesn't satisfy NIST requirements. The authors of ref. [12] also extended the aforementioned attack to Remus-N1. ICE2 (used in Remus-N2) provides higher security of the form $DT/2^{2k}$, but it uses an

auxiliary key (can be viewed as a combination of XEX and ICE1). This costs some additional state to hold this auxiliary key. This motivates the following question.

Can we have an instantiation of TBC based on block cipher without using any auxiliary key but still provide satisfactory security (for instance, up to the NIST desired level)?

At a glance, it seems impossible to design such a TBC. However, we show that such security is possible to achieve for the AEAD mode based on a TBC, which does not use any auxiliary key. Clearly, we must have different reductions than what we usually have (as mentioned in equations (1.1)–(1.2)). The distinguishing attack for Remus-N1 [7] requires fixing the associated data block as it is going to be the input of the underlying block cipher. This can be avoided if the first block is defined to be nonce. This principle is adapted in mixFeed [6]. Both mixFeed [6] and Remus [7] are similar in nature (as described earlier) with two main differences. First, they use different feedback functions that do not affect the security as such. The second difference is that unlike Remus, mixFeed processes the nonce before processing the associated data and hence uses one extra TBC call. In the nonce-respecting model, the inputs of the block cipher vary in the case of mixFeed. Hence, the attack due to ref. [12] can be avoided.

1.1 Our contribution

We formalize the TBC-based AEAD mode, which is used in mixFeed, and we call it mF. In this article, we reduce the security of the mF mode in terms of the security of the underlying TBC against two newly introduced input-restricted adversaries, namely, (1) the μ -respecting TPRP adversary and (2) (μ, λ) -multi-commitment prediction adversary. We note that these two notions are non-standard but weaker than standard TPRP security notions in the sense that an adversary is μ -respecting if it can make at most μ queries with the same input. In a nonce-respecting AEAD scheme where the nonce is processed in the beginning, since the initial TBC input is not repeated, and the other TBC inputs depend on their previous TBC outputs, we can consider the security of the underlying TBC in terms of μ -respecting adversaries with small μ . In the case of mF mode in nonce-respecting setup, we can choose μ to be about n to achieve the following bounds.

$$\mathbf{Adv}_{\text{mF}}^{\text{priv}}(T, D) \leq \mathbf{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(T, D) + O\left(\frac{\mu^2 D}{2^n}\right), \quad (1.3)$$

$$\mathbf{Adv}_{\text{mF}}^{\text{forge}}(T, D) \leq \mathbf{Adv}_{\tilde{E}}^{(\mu, D)\text{-mcp}}(T, D) + 2 \cdot \mathbf{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(T, D) + O\left(\frac{\mu^2 D}{2^n}\right) + \frac{2D}{2^{\frac{n}{2}}}. \quad (1.4)$$

where the new security advantage terms $\mathbf{Adv}_{\tilde{E}}^{\mu\text{-tprp}}$ and $\mathbf{Adv}_{\tilde{E}}^{(\mu, D)\text{-mcp}}$, respectively, denote the adversarial advantage of any μ -respecting TPRP adversary and any (μ, D) -multicommitment prediction adversary. For more detailed definitions, see Section 2.2.

We now study these two security notions for two instantiations of TBC. mixFeed is one such instantiation. The designers of mixFeed claimed the security of the scheme under the assumption that the AES key scheduling algorithm has a small number of short permutation cycles, and hence, the probability of finding a key in those cycles is also small. Khairallah [13] observed that if this assumption is violated, it may lead to weak-key attacks on mixFeed. Later, Leurent and Pernot [14] confirmed this observation by explicitly finding a large number of short cycles in the AES key scheduling algorithm. In this article, we try to interpret this weakness in terms of our general notations and conclude that the weakness is only due to the use of the AES key schedule in the key updation function (KUF). Our second instantiation of TBC based on block cipher using primitive element multiplication (we call the overall AEAD mF_{prim}) achieves the bounds,

$$\mathbf{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(T, D) \approx \frac{\mu T}{2^n}. \quad (1.5)$$

$$\mathbf{Adv}_{\tilde{E}}^{(\mu, D)\text{-mcp}}(T, D) \approx \frac{\mu T}{2^n} + \frac{D}{2^{\frac{n}{2}}}. \quad (1.6)$$

Plugging these results in our previous results, we obtain that the $\mathbf{mF}_{\text{prim}}$ mode is well secured within the NIST requirements.

1.2 Organization of the article

In Section 2, we define some existing and newly introduced security definitions associated with AEAD modes and TBCs. In Section 3, we introduce the TBC-based AEAD scheme called $\mathbf{mF}$. In Section 4, we reduce the security of the $\mathbf{mF}$ mode to the security of the underlying TBC against the newly introduced TBC security games defined in Section 2.2. In Section 5, we define a new TBC construction using a block cipher and a KUF and upper bound the advantages of any adversary playing those new TBC security games against it. In Section 6, we consider the $\mathbf{mF}$ mode under this new TBC and derive an upper bound on the security of such a mode. Further, we inspect the weakness of the $\mathbf{mixFeed}$ mode as an explicit instantiation of the $\mathbf{mF}$ mode and try to interpret it in our notation. In Section 7, we show that this weakness of $\mathbf{mixFeed}$ doesn't weaken the security of the $\mathbf{mF}$ mode in general by constructing an explicit instantiation called the $\mathbf{mF}_{\text{prim}}$ mode, which is well secured within the NIST-specified parameters. Finally, in Section 8, we make a theoretical comparison between the $\mathbf{mF}_{\text{prim}}$ mode of AEAD and some other existing TBC-based lightweight AEAD schemes.

2 Security definitions of AEAD and TBC

2.1 Security definitions of AEAD

In this section, we define some well-known security definitions of AEAD modes. We call an AEAD adversary nonce respecting when it doesn't make more than one encryption query with the same nonce. An authenticated encryption scheme with associated data functionality, or AEAD in short, is a tuple of deterministic algorithms $F = (\mathcal{E}, \mathcal{D})$, defined over the *key space* $\mathcal{K}$, *nonce space* $\mathcal{N}$, *associated data space* $\mathcal{A}$, *message space* $\mathcal{M}$, *ciphertext space* $\mathcal{C}$, and *tag space* $\mathcal{T}$, where:

$$\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M} \rightarrow \mathcal{C} \times \mathcal{T} \quad \text{and} \quad \mathcal{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{C} \times \mathcal{T} \rightarrow \mathcal{M} \cup \{\perp\}.$$

Here, $\mathcal{E}$ and $\mathcal{D}$ are called the encryption and decryption algorithms, respectively, of F . Further, it is required that $\mathcal{D}(K, N, A, \mathcal{E}(K, N, A, M)) = M$ for any $(K, N, A, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M}$. For all key $K \in \mathcal{K}$, we write $\mathcal{E}_K(\cdot)$ and $\mathcal{D}_K(\cdot)$ to denote $\mathcal{E}(K, \cdot)$ and $\mathcal{D}(K, \cdot)$, respectively.

2.1.1 Privacy

Given an adversary $\mathcal{A}$, we define the *privacy advantage* [15] of $\mathcal{A}$ against F as $\mathbf{Adv}_F^{\text{priv}}(\mathcal{A}) = |\Pr[\mathcal{A}^{\mathcal{E}_K} = 1] - \Pr[\mathcal{A}^{\$} = 1]|$, where $\$$ returns a random output string of the same length as the output size of $\mathcal{E}_K$. The *privacy advantage* of the F_K is defined as follows:

$$\mathbf{Adv}_F^{\text{priv}}(q, \sigma, t) = \max_{\mathcal{A}} \mathbf{Adv}_F^{\text{priv}}(\mathcal{A}),$$

where the maximum is taken over all the nonce-respecting adversaries $\mathcal{A}$ running in time t and making at most q encryption queries with a total number of blocks in all the queries being at most σ .

2.1.2 Forgery

We say that a nonce-respecting oracle adversary $\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K}$ forges F , if $\mathcal{A}$ is able to make a fresh and valid query (N, A, C, T) to $\mathcal{D}_K$. By fresh query, we mean that the adversary does not make any previous query (N, A, M) to $\mathcal{E}_K$ such that $\mathcal{E}_K(N, A, M) = (C, T)$. We say a decryption query is valid, if $\mathcal{D}_K(N, A, C, T) \neq \perp$. The *forging advantage* [15] of an adversary $\mathcal{A}$ is written as follows:

$$\mathbf{Adv}_F^{\text{forge}}(\mathcal{A}) = \Pr[\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K} \text{ forges}]$$

and we write

$$\mathbf{Adv}_F^{\text{forge}}(q, \sigma, t) = \max_{\mathcal{A}} \mathbf{Adv}_F^{\text{forge}}(\mathcal{A}),$$

where the maximum is taken over all adversary $\mathcal{A}$ running in time t , making at most q_e nonce-respecting encryption queries with maximum σ_e blocks and making at most q_d decryption queries with maximum σ_d blocks. We define $q = q_e + q_d$ and $\sigma = \sigma_e + \sigma_d$. Note that the decryption queries are not necessarily nonce respecting, i.e., nonce can be repeated in the decryption queries, and an encryption query and a decryption query can use the same nonce. However, all nonces used in encryption queries are distinct.

2.2 Security definitions of TBC

Let $\tilde{E}$ be an n -bit TBC with tweak space $\mathcal{T}$. The *TPRP advantage* [5] of $\tilde{E}$ against an oracle adversary $\mathcal{A}$ is defined as follows:

$$\mathbf{Adv}_{\tilde{E}}^{\text{tprp}}(\mathcal{A}) = |\Pr[\mathcal{A}^{\tilde{E}_K} = 1] - \Pr[\mathcal{A}^{\tilde{\Pi}} = 1]|,$$

where $\tilde{\Pi}$ is chosen uniformly from the set of all functions $\tilde{\pi} : \mathcal{T} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$, where, for every $tw \in \mathcal{T}$, $\tilde{\pi}(tw, \cdot)$ is a permutation on $\{0, 1\}^n$. We call $\tilde{\Pi}$ a tweakable random permutation. We write $\mathbf{Adv}_{\tilde{E}}^{\text{tprp}}(q, t) = \max_{\mathcal{A}} \mathbf{Adv}_{\tilde{E}}^{\text{tprp}}(\mathcal{A})$ where maximum is taken over all adversaries $\mathcal{A}$ running in time t making at most q queries.

Next, we define two new security definitions for a given TBC.

2.2.1 μ -respecting TPRP security

Let μ be a positive integer. We define μ -TPRP advantage of $\tilde{E}$ to be $\mathbf{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(q, t) = \max_{\mathcal{A}} \mathbf{Adv}_{\tilde{E}}^{\text{tprp}}(\mathcal{A})$, where the maximum is taken over all μ -respecting adversaries $\mathcal{A}$ (i.e., the number of queries (tw, X) by $\mathcal{A}$ with a same plaintext input X is at most μ) running in time t . When the TBC is instantiated in the ideal cipher model, the time parameter t denotes the number of ideal cipher calls.

2.2.2 Multi-commitment prediction

Let $\tilde{E}$ be a TBC. Let $\mathcal{A}$ be an adversary, which has oracle access to an n -bit TBC $\tilde{E}$ with a tweak space $\mathcal{T}$ in the first phase.

PHASE 2:

- (i) After all the queries of the first phase are done, it makes at most λ commitments of the form (tw_i, x_i, y_i) , where $x_i, y_i \in \{0, 1\}^n$, $tw_i \in \mathcal{T}$.
- (ii) $\mathcal{A}$ makes prediction queries of the form (tw'_j, X_j) such that (tw'_j, X_j) are fresh, i.e., $\forall j, (tw'_j, X_j)$ has never been queried before (Figure 1).

We say that any adversary $\mathcal{A}$ wins the λ -multi-commitment-prediction game if for some prediction query tuple (tw'_j, X_j) there exists a commitment tuple (tw_i, x_i, y_i) such that

$$tw_i = tw'_j; x_i = \lceil X_j \rceil_{\frac{n}{2}}; \lfloor \tilde{E}_K(tw_i, X_j) \rfloor_{\frac{n}{2}} = y_i.$$

The λ -multi-commitment-predicting advantage of an adversary $\mathcal{A}$ is defined as follows:

$$\text{Adv}_E^{\lambda\text{-mcp}}(\mathcal{A}) = \Pr[\mathcal{A}^{\tilde{E}} \text{ wins the } \lambda\text{-multi-commitment-prediction game}],$$

and we write,

$$\text{Adv}_E^{\lambda\text{-mcp}}(q, t) = \max_{\mathcal{A}} \text{Adv}_E^{\lambda\text{-mcp}}(\mathcal{A}),$$

where maximum is taken over all adversaries $\mathcal{A}$ running in time t making at most q queries.

We define (μ, λ) -mcp advantage of $\mathcal{A}$ to be

$$\text{Adv}_E^{(\mu, \lambda)\text{-mcp}}(q, t) = \max_{\mathcal{A}} \text{Adv}_E^{\lambda\text{-mcp}}(\mathcal{A})$$

where the maximum is taken over all adversaries as defined earlier with the additional restriction that they make μ -respecting queries in the first phase of the game.

We would like to note that in the ideal cipher model, the (μ, λ) -multi-commitment prediction security is defined in the same way as mentioned earlier with an additional restriction that the adversary doesn't make any primitive calls to E in the phase 2.

2.3 Multi collision

We say that an adversary $\mathcal{A}$ with oracle access to O produces a μ -multicollision if it makes μ distinct queries $x_1, \dots, x_\mu$ to O , for which all responses are identical. The μ -multicollision-advantage of the adversary $\mathcal{A}$ is defined as follows:

$$\text{Adv}_O^{\mu\text{-mcoll}}(\mathcal{A}) = \Pr[\mathcal{A}^O \text{ produces } \mu\text{-multicollision}]$$

Figure 1: Phase 2 of the (λ, μ) -mcp game between $\mathcal{A}$ and $\mathcal{CH}$. The phase 1 $\mathcal{CH}$ queries are responded similarly to as in the case of μ -TPRP game. For phase 2 queries, the μ -restriction is lifted. Note that the phase 2 queries and predictions can be done in any order. The only condition is that each of the predictions of the form (tw'_j, X'_j) must be fresh, i.e., it has not been queried earlier.

Figure 2: The FB^+ function in mF. pad is 0^*1 padding.

and $\text{Adv}_O^{\mu\text{-mcoll}}(q) = \max_{\mathcal{A}} \text{Adv}_O^{\mu\text{-mcoll}}(\mathcal{A})$, where maximum is taken over all adversaries $\mathcal{A}$ making at most q queries.

3 The mF mode of AEAD

We start by defining the positive feedback function FB^+ , which takes a chain input Y of size n -bits and a data input M of size less than or equal to n -bits to generate a data output C of size $|M|$ -bits and a chain output of size n . The negative feedback function can be described in a similar way. The general description of the feedback functions can be understood from Figures 2 and 3. For the case when the data input M has length n , the feedback functions can be described in a much simpler way. If $|M| = n$, then $FB^+(Y, M) = (X, C)$, where $C = Y \oplus M$ and $X = \lceil C \rceil_{n/2} \parallel \lfloor M \rfloor_{n/2}$. Similarly, if $|C| = n$, then $FB^-(Y, C) = (X, M)$, where $M = Y \oplus C$ and $X = \lceil C \rceil_{n/2} \parallel \lfloor M \rfloor_{n/2}$.

Let $\tilde{E}$ be a TBC with state size n and tweak-size $t > n - 8$. We define the mF mode of AEAD using this TBC as follows. Given any data D , we define $d := \left\lfloor \frac{|D|}{n} \right\rfloor$. We parse the data D into d parts of n bit data blocks.

In notation $(D_d, \dots, D_1) \stackrel{n}{\leftarrow} D$, where $|D_d| = \begin{cases} n & \text{if } n \mid |D| \\ r & \text{if } \exists r > 0 \text{ s.t. } |D| \equiv r \pmod{n} \end{cases}$.

Given any data $(N, A, M) \in \{0, 1\}^{n-8} \parallel \{0, 1\}^* \{0, 1\}^*$ and distinct but predefined $\{a_1, \dots, a_6\}$, we define (a, δ_A) and (m, δ_M) depending on $|A|$ and $|M|$ using the Fmt function as described in Algorithm 5. We restrict the values of a, m such that $a + m + 2 \leq 2^{t-n+8}$. With this setup, the mF mode encryption with the secret key K , simply outputs $\tilde{E}_K(0^t, N \parallel 0^6 10)$ as the tag if both $a, m = 0$. Else it sets $N' = N \parallel 0^7 1$ if $a = 0$ and $N' = N \parallel 0^8$ if $a \neq 0$. It takes $((N', 0), N \parallel 0^8)$ as the first TBC input to generate Y_0 .

Associated data processing: It parses $(A_a, \dots, A_1) \stackrel{n}{\leftarrow} A$. For each $i \in [0, a - 1]$, it evaluates $FB^+(Y_i, A_{i+1})$ to generate $(X_{i+1}, *)$ and use $((N', i), X_{i+1})$ as the next TBC input to generate Y_{i+1} . Finally, it makes another TBC call with input $((N', a + 1), Y_a \oplus \delta_A)$ to generate Y_{a+1} . It outputs Y_{a+1} as tag, if $m = 0$.

Message processing: It parses $(M_m, \dots, M_1) \stackrel{n}{\leftarrow} M$. For each $i \in [a + 1, l]$ it evaluates $FB^+(Y_i, M_{i-a})$ to generate (X_{i+1}, C_{i-a}) and use $((N', i), X_{i+1})$ as the next TBC input to generate Y_{i+1} . Finally it makes another TBC call with input $((N', l + 2), Y_{l+1} \oplus \delta_M)$ to generate the tag. It outputs $(C_m, \dots, C_1)$ as the ciphertext.

Let pad be 0^*1 padding function. $\perp$ represents invalid. $P_1? a_1 : a_2$ evaluates to a_1 if P_1 is true and a_2 otherwise. $P_1 \& P_2? a_1 : a_2 : a_3 : a_4$ evaluates to a_1 if both P_1 and P_2 are true, to a_2 if only P_1 is true, to a_3 if only P_2 is true and to a_4 if none of P_1, P_2 are true. $\text{Feed}(*, *, \text{dir}) := \begin{cases} FB^+ & \text{if dir} = + \\ FB^- & \text{if dir} = - \end{cases}$. Then, with this notations,

mF mode of AEAD can be best understood from Figure 4 and 5.

Figure 3: The FB^- function in mF. pad is 0^*1^* padding.

Figure 4: Block diagram for mF encryption. Here, $N' = N||x$, where $x = 0^8/0^7 1$ depending on the condition that $(a \neq 0)$ or $(a = 0 \ \& \ m \neq 0)$ respectively. We define $l = a + m$. δ_A, δ_M are as defined in Figure 5.

4 Security reductions of mF

Here, we give upper bounds on the *privacy advantage* and *forging advantage* of mF against any adversary $\mathcal{B}$. For notational reference, see Figure 4.

4.1 Privacy

Theorem 4.1. For any privacy adversary $\mathcal{B}$ of mF, there is an μ -TPRP adversary $\mathcal{A}$ of $\tilde{E}$, such that

<pre> 1: function mF[E].enc(K, N, A, M) 2: $((a, \delta_A), (m, \delta_M)) \leftarrow \text{Fmt}(A, M)$ 3: if $a = 0, m = 0$ then 4: $T \leftarrow \tilde{E}_K(0^t, N 0^6 10)$ 5: return (λ, T) 6: else if $a = 0$ then $N' \leftarrow N 0^7 1$ 7: else $N' \leftarrow N 0^8$ 8: $T \leftarrow \tilde{E}_K((N', 0), N 0^8)$ 9: $C \leftarrow \lambda$ 10: if $a \neq 0$ then 11: $(*, T) \leftarrow \text{proc_txt}(N', 0, K, T, A, \delta_A, +)$ 12: if $m \neq 0$ then 13: $(C, T) \leftarrow \text{proc_txt}(N', a + 1, K, T, M, \delta_M, +)$ 14: return (C, T) </pre>	<pre> 1: function Fmt(A, M) 2: $(A_{a-1}, \dots, A_0) \xleftarrow{r} A$ 3: $(M_{m-1}, \dots, M_0) \xleftarrow{r} M$ 4: $\delta_A \leftarrow (n \mid A_{a-1}) \ \& \ (m = 0)? a_1 : a_2 : a_3 : a_4$ 5: $\delta_M \leftarrow (n \mid M_{m-1})? a_6 : a_5$ 6: return $((a, \delta_A), (m, \delta_M))$ </pre>
<pre> 15: function mF[E].dec(K, N, A, C, T) 16: $((a, \delta_A), (m, \delta_C)) \leftarrow \text{Fmt}(A, C)$ 17: if $a = 0, m = 0$ then 18: $(T', *) \leftarrow E_K(N 0^6 10)$ 19: return $(T = T')? \top : \perp$ 20: else if $a = 0$ then $N' \leftarrow N 0^7 1$ 21: else $N' \leftarrow N 0^8$ 22: $T \leftarrow \tilde{E}_K((N', 0), N 0^8)$ 23: $C \leftarrow \lambda$ 24: if $a \neq 0$ then 25: $(*, T') \leftarrow \text{proc_txt}(N', 0, K, T', A, \delta_A, +)$ 26: if $m \neq 0$ then 27: $(M, T') \leftarrow \text{proc_txt}(N', a + 1, K, T', C, \delta_C, -)$ 28: if $T \neq T'$ then 29: return $\perp$ 30: else 31: return $(M, \top)$ </pre>	<pre> 7: function proc_txt(N, l, K, Y_0, D, \delta_D, dir) 8: $(D_d, \dots, D_1) \xleftarrow{r} D$ 9: for $i = 1$ to d do 10: $(X_i, D'_i) \leftarrow \text{Feed}(Y_{i-1}, D_i, \text{dir})$ 11: $Y_i \leftarrow \tilde{E}_K((N, l + i), X_i)$ 12: $X_{d+1} \leftarrow Y_d \oplus 0^{n-4} \delta_D$ 13: $Y_{d+1} \leftarrow \tilde{E}_K((N, l + d + 1), X_{d+1})$ 14: return (D', Y_{d+1}) </pre>
	<pre> 15: function Feed(Y, D, dir) 16: $D' \leftarrow D \oplus \lfloor Y \rfloor_{ D }$ 17: if dir = "+" then 18: $B \leftarrow \lceil \text{pad}(D') \rceil_{n/2} \lfloor \text{pad}(D) \rfloor_{n/2}$ 19: if dir = "-" then 20: $B \leftarrow \lceil \text{pad}(D) \rceil_{n/2} \lfloor \text{pad}(D') \rfloor_{n/2}$ 21: $X \leftarrow B \oplus Y$ 22: return (X, D') </pre>

Figure 5: Algorithm defining the mF mode.

$$\mathbf{Adv}_{\text{mF}}^{\text{priv}}(\mathcal{B}) \leq \mathbf{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(\mathcal{A}) + \sigma \left(1 + \frac{(\mu+1)^2}{2^n} \right) \left(\frac{\sigma}{2^{\frac{n}{2}}} \right)^\mu,$$

where σ is the total number of queries by $\mathcal{A}$.

Proof. Note that mF is the mode based on a TBC $\tilde{E}_K$. If we replace $\tilde{E}_K$ by an n -bit tweakable random permutation P with same tweak space, we denote the construction as mF_P . From the construction it is easy to see that all tweaks used in the tweakable random permutation while we execute nonce-respecting queries are distinct. Hence, all output bits of mF_P are random, so it is equivalent to the oracle $\$$. So, $\mathbf{Adv}_{\text{mF}}^{\text{priv}}(\mathcal{B}) = \Pr[\mathcal{B}^{\text{mF}_P} = 1] - \Pr[\mathcal{B}^{\text{mF}_{\tilde{E}_K}} = 1]$.

By using straightforward reduction, one can construct an adversary $\mathcal{A}'$, which mainly simulates the mode mF_O , where O (which is either P or $\tilde{E}_K$) is the challenge oracle of $\mathcal{A}'$. Clearly, $\mathbf{Adv}_{\text{mF}}^{\text{priv}}(\mathcal{B}) \leq \mathbf{Adv}_{\tilde{E}}^{\text{tprp}}(\mathcal{A}')$. However, $\mathcal{A}'$ does not necessarily follow μ -input-respecting for small μ . So we follow a slightly different strategy to define $\mathcal{A}$. It is basically same as $\mathcal{A}'$, except that it aborts and returns 0 whenever it is going to violate μ -input-restriction. More precisely, it maintains a list of all queries (tw_i, X_i) to its challenger. If for some i , there exists $\mu+1$ number of $j < i$ with $X_j = X_i$, it aborts (instead of making the query) and returns zero. When it does not abort, it returns whatever $\mathcal{B}$ returns. Now,

$$\begin{aligned} & \Pr[\mathcal{B}^{\text{mF}_P} = 1] - \Pr[\mathcal{B}^{\text{mF}_{\tilde{E}_K}} = 1] \\ & \leq \Pr[\mathcal{B}^{\text{mF}_P} = 1 \cap \mathcal{A} \text{ doesn't abort}] + \Pr[\mathcal{A}^P \text{ aborts}] - \Pr[\mathcal{B}^{\text{mF}_{\tilde{E}_K}} = 1 \cap \mathcal{A} \text{ doesn't abort}] \\ & \leq_{(1)} \left(\Pr[\mathcal{A}^P = 1] - \Pr[\mathcal{A}^{\text{mF}_{\tilde{E}_K}} = 1] \right) + \Pr[\mathcal{A}^P \text{ aborts}] \\ & \leq_{(2)} \mathbf{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(\mathcal{A}) + \sigma \left(1 + \frac{(\mu+1)^2}{2^n} \right) \left(\frac{\sigma}{2^{\frac{n}{2}}} \right)^\mu. \end{aligned}$$

The inequality (1) follows from the definition that $\mathcal{A}$ returns 1 if and only if it does not abort and $\mathcal{B}$ returns 1. We now bound $\Pr[\mathcal{A}^P \text{ aborts}] \leq \sigma \left(1 + \frac{(\mu+1)^2}{2^n} \right) \left(\frac{\sigma}{2^{\frac{n}{2}}} \right)^\mu$, which justifies inequality (2) mentioned earlier.

Consider the event that $\mathcal{A}^P$ aborts, i.e., $\exists \{i_1, \dots, i_{\mu+1}\} \in [1, \sigma]$ such that $\mathcal{A}$ needs to make $\mu+1$ queries of the form (tw_{i_j}, X_{i_j}) to P such that $X_{i_j} = X_{i_{j'}} \forall j, j' \in [1, \mu+1]$. Now, note that for all such queries, if the previous query by $\mathcal{A}$ is of the form $(tw_{i_{j-1}}, X_{i_{j-1}})$ and it received the output $Y_{i_{j-1}}$, then

$$\lfloor X_{i_j} \rfloor_{\frac{n}{2}} = \lfloor Y_{i_j} \oplus C_{i_j} \rfloor_{\frac{n}{2}}$$

for some known $C_{i_1}, \dots, C_{i_{\mu+1}}$. Now consider an oracle O_P , which takes input of the form (tw_i, X_i, C_i) and outputs $z_i = \lfloor Y_i \oplus C_i \rfloor_{\frac{n}{2}}$ as response, where $Y_i = P(tw_i, X_i)$.

Then, clearly,

$$\Pr[\mathcal{A}^P \text{ aborts}] \leq \mathbf{Adv}_{O_P}^{\mu+1\text{-mcoll}}(\sigma).$$

Finally, to bound $\mathbf{Adv}_{O_P}^{\mu+1\text{-mcoll}}(\sigma)$, let $\omega_d = ((N_i, j_i), X_i, z_i)_{i \in \mathcal{E}}$ be the online transcript of any adversary playing μ -mcoll game with O_P .

The μ -multi collision occurs if $\exists i_1, \dots, i_\mu \in [1, \sigma]$ such that $z_{i_k} = z_{i_l}$ for all $k, l \in [1, \mu]$.

Note that the probability of μ -multicollision is highest when the tweak is the same for all the queries.

In that case, for a given $x \in \{0, 1\}^{\frac{n}{2}}$ and fixed $i_k \in [1, \sigma]$ number of possible tuples of (Y_{i_k}, C_{i_k}) such that $z_{i_k} = \lfloor Y_{i_k} \oplus C_{i_k} \rfloor_{\frac{n}{2}} = x$ is bounded by $2^{\frac{n}{2}}$. Hence, varying over all $i_1, \dots, i_\mu \in [1, \sigma]$, we have a number of possible tuples $(Y_{i_1}, C_{i_1}), \dots, (Y_{i_\mu}, C_{i_\mu})$ such that $z_{i_k} = \lfloor Y_{i_k} \oplus C_{i_k} \rfloor_{\frac{n}{2}} = x \forall k \in [1, \mu]$ is bounded by $2^{\frac{\mu n}{2}}$. $\square$

Varying over all $x \in \{0, 1\}^{\frac{n}{2}}$ and for all combination of $i_1, \dots, i_\mu \in [1, \sigma]$, we have number of ways in which μ -multi collision occurs is at most $\binom{\sigma}{\mu} 2^{\frac{(\mu+1)n}{2}}$.

Hence, we have

Lemma 4.2.

$$\mathbf{Adv}_{O_p}^{\mu\text{-mcoll}}(\sigma) = \Pr[\mu\text{-mcoll}] \leq \frac{\binom{\sigma}{\mu} 2^{\frac{(\mu+1)n}{2}}}{(2^n)^\mu} \leq \sigma \left(1 + \frac{\mu^2}{2^n}\right) \left(\frac{\sigma}{2^{\frac{n}{2}}}\right)^{\mu-1}.$$

4.2 Forgery

Define an oracle $O_{\tilde{E}_K}$, which takes a query input of the form (tw, X, C) and returns

$$X' = C \oplus \left(0^{\frac{n}{2}} \parallel \tilde{E}_K(tw, X) \right)_{\frac{n}{2}}.$$

We can similarly define O_P , where the TBC mentioned earlier is replaced by tweakable random permutation P . For any $(\mu + 1)$ -multicollision $(\mu + 1)$ -input-restricting adversary $\mathcal{C}$ with oracle access to $O_{\tilde{E}_K}$, there is an $(\mu + 1)$ -multicollision adversary $\mathcal{C}'$ with oracle access to O_P , such that

$$\mathbf{Adv}_{\tilde{E}}^{(\mu+1)\text{-mcoll}}(\mathcal{C}) \leq \mathbf{Adv}_{\tilde{E}}^{\mu\text{-trpr}}(\mathcal{C}') + \sigma \left(1 + \frac{(\mu + 1)^2}{2^n}\right) \left(\frac{\sigma}{2^{\frac{n}{2}}}\right)^\mu.$$

This follows from the standard reduction and Lemma 4.2.

Theorem 4.3. For any nonce-respecting forging adversary $\mathcal{B}$ of mF making q_e encryption queries with σ_e encryption query blocks, q_d decryption queries with σ_d decryption query blocks, there is (i) (μ, σ_d) -mcp adversary $\mathcal{A}$ of $\tilde{E}$, and (ii) $(\mu + 1)$ -multicollision adversary $\mathcal{C}$ with oracle access to $O_{\tilde{E}_K}$ (as defined earlier), such that

$$\mathbf{Adv}_{\text{mF}}^{\text{forge}}(\mathcal{B}) \leq \mathbf{Adv}_{\tilde{E}}^{(\mu, \sigma_d)\text{-mcp}}(\mathcal{A}) + 2 \cdot \mathbf{Adv}_{\tilde{E}}^{\mu\text{-trpr}}(\mathcal{C}) + \sigma \left(1 + \frac{(\mu + 1)^2}{2^n}\right) \left(\frac{\sigma}{2^{\frac{n}{2}}}\right)^\mu + \frac{2\sigma_e}{2^{\frac{n}{2}}}.$$

Let $\mathcal{B}$ be any forging adversary of mF. Suppose $\mathcal{B}$ makes q_e encryption queries with σ_e encryption query blocks and q_d forging attempts with effectively σ_d encryption blocks. We construct a (μ, σ_d) -mcp adversary $\mathcal{A}$, which uses $\mathcal{B}$ to win the (μ, σ_d) -multi-commitment-prediction game of $\tilde{E}$.

4.2.1 The reduction game

Let $\mathcal{CH}$ be a (μ, λ) -mcp challenger. $\mathcal{A}$ acts as a forgery challenger for $\mathcal{B}$, as follows:

Phase 1:

- (I) Whenever $\mathcal{B}$ sends an encryption query of the form $(N^i, A^i, M^i)_{i \in [1, q_e]}$,
 - (A) $\mathcal{A}$ responds to the query by computing (C^i, T^i) by making the required $\tilde{E}_K$ queries to $\mathcal{CH}$.
 - (B) In the previous step, $\mathcal{A}$ always follows the restriction that no more than μ queries to $\tilde{E}$ have the same input. Otherwise, it aborts.
- (II) For every decryption query of the form $(N^{*j}, A^{*j}, C^{*j}, T^{*j})_{j \in [1, q_d]}$, $\mathcal{A}$ simply responds it with $\perp$.
- (III) When all the encryption and decryption queries by $\mathcal{B}$ have been responded, $\mathcal{A}$ revisits all the decryption queries made by $\mathcal{B}$. For each $j \in [1, q_d]$, $\mathcal{A}$ proceeds as follows:
 - (A) $\mathcal{A}$ checks if $\mathcal{B}$ has previously made any encryption query (N^i, A^i, M^i) and received output of the form (C^i, T^i) such that $N^i = N^{*j}$ and defines an integer p_j as follows:
 - (a) if there doesn't exist any encryption query (N^i, A^i, M^i) from $\mathcal{B}$ such that $N^i = N^{*j}$, then $\mathcal{A}$ sets $p_j = -1$.
 - (b) Else if $\exists (N^i, A^i, M^i)$ such that $N^i = N^{*j}$ but $T^i \neq T^{*j}$ or $l_i < l_j^*$, then $\mathcal{A}$ sets $p_j = -1$.
 - (c) Else if $\exists (N^i, A^i, M^i)$ such that $N^i = N^{*j}$ but $l_i > l_j^*$ and $T^{*j} \neq Y_{l_j^*+2}^i$, then $\mathcal{A}$ sets $p_j = -1$.
 - (d) Else if $\exists (N^i, A^i, M^i)$ such that $N^i = N^{*j}$ but $l_i > l_j^*$ and $T^{*j} = Y_{l_j^*+2}^i$, then $\mathcal{A}$ sets $p_j = 0$.

- (e) Else if $p'_j \in \mathbb{Z}_{\geq 0}$ be such that $\text{pad}(C_{m_j^* - k}^{*j}) = \text{pad}(C_{m_i - k}^i), \forall k \in [0, p'_j)$ but $\text{pad}(C_{m_j^* - p'_j}^{*j}) \neq \text{pad}(C_{m_i - p'_j}^i)$, then
- (i) $\mathcal{A}$ defines $p_j = p'_j + 1$, if $\lceil \text{pad}(C_{m_j^* - p'_j}^{*j}) \rceil_{\frac{n}{2}} = \lceil \text{pad}(C_{m_i - p'_j}^i) \rceil_{\frac{n}{2}}$. Otherwise, it defines $p_j = p'_j + 2$.
- (ii) $\mathcal{A}$ defines

$$\Delta^j := \lfloor \text{pad}(C_{m_j^* - p_j + 1}^{*j}) \rfloor_{\frac{n}{2}} \oplus \lfloor \text{pad}(C_{m_i - p_j + 1}^i) \rfloor_{\frac{n}{2}}.$$

- (B) If $p_j = -1, 0$; $\mathcal{A}$ computes Y_k^{*j} for all $k \in [0, l_j^* - p_j]$ and else computes Y_k^{*j} for all $k \in [0, l_j^* - p_j + 1]$ with the help of $\mathcal{CH}$ following the restriction that no more than μ queries to $\tilde{E}$ have the same input. In that case $\mathcal{A}$ aborts.

Remark 4.4. If there exists a common prefix between (N^i, A^i, C^i) and (N^{*j}, A^{*j}, C^{*j}) , then $\mathcal{A}$ already has computed up to the common prefix length during encryption query and thus need not send any new encryption query to $\mathcal{CH}$ for computation up to that point.

Phase 2 (commitment):

For each $j \in [1, q_d]$,

- (I) If $p_j = -1$, then,

- (A) Note that $\mathcal{A}$ knows $Y_{l_j^* + 1}^{*j}$ from Phase 1.
- (B) $\mathcal{A}$ sets commitment of the form $((N^{*j}, l_j^* + 2), \lceil Y_{l_j^* + 1}^{*j} \rceil_{\frac{n}{2}}, \lfloor T^{*j} \rfloor_{\frac{n}{2}})$.

- (II) If $p_j = 0$, then,

- (A) $\mathcal{A}$ sets $\lfloor Y_{l_j^* + 1}^{*j} \rfloor_{\frac{n}{2}} = \lfloor Y_{l_j^* + 1}^i \oplus D_{l_j^* + 1}^i \oplus \delta_{M^{*j}} \rfloor_{\frac{n}{2}}$, where

$$D_{l_j^* + 1}^i = \begin{cases} Y_{l_j^* + 1}^i \oplus A_{l_j^* + 2}^i & \text{if } l_j^* < a_i \\ \delta_{A^i} & \text{if } l_j^* = a_i \\ C_{l_j^* - a_i + 1}^i & \text{if } l_j^* > a_i. \end{cases}$$

- (B) $\mathcal{A}$ sets commitment of the form $((N^{*j}, l_j^* + 1), \lceil C_{m_j^*}^{*j} \rceil_{\frac{n}{2}}, \lfloor Y_{l_j^* + 1}^{*j} \rfloor_{\frac{n}{2}})$.

- (III) If $p_j \neq 0, -1$, then $\mathcal{A}$ makes p_j commitments of the form

$$(tw_k^{*j}, x_k^{*j}, y_k^{*j})_{k \in [l_j^* - p_j + 2, l_j^* + 1]},$$

where

$$tw_k^{*j} = (N^{*j}, k); x_k^{*j} = \lceil C_{m_j^* - p_j + 1}^{*j} \rceil_{\frac{n}{2}};$$

$$y_k^{*j} = \begin{cases} \lfloor Y_k^i \rfloor_{\frac{n}{2}} \oplus \Delta^j & \text{if } k = l_j^* - p_j + 2 \\ \lfloor Y_k^i \rfloor_{\frac{n}{2}} & \text{Otherwise.} \end{cases}$$

Phase 2 (prediction):

For each $j \in [1, q_d]$

- (I) If $p_j = -1$, then

- (A) It calculates $X_{l_j^* + 2}^{*j} = Y_{l_j^* + 1}^{*j} \oplus \delta_{M^{*j}}$.
- (B) It sends prediction query of the form $((N^{*j}, l_j^* + 2), X_{l_j^* + 2}^{*j})$.

- (II) If $p_j = 0$, then

- (A) Note that $\mathcal{A}$ knows $Y_{l_j^*}^{*j}$ from Phase 1.
- (B) $\mathcal{A}$ then sets $X_{l_j^* + 1}^{*j} = (O_{\frac{n}{2}} \parallel \lfloor Y_{l_j^*}^{*j} \rfloor_{\frac{n}{2}}) \oplus C_{m_j^*}^{*j}$.
- (C) Finally $\mathcal{A}$ send $((N^{*j}, l_j^* + 1), X_{l_j^* + 1}^{*j})$ as a prediction query.

(III) If $p_j \neq 0, -1$, then

- (A) Note that, $\mathcal{A}$ knows $Y_{l_j^* - p_j + 1}^{*j}$ from Phase 1.
- (B) for $k = l_j^* - p_j + 2$ to $l_j^* + 1$,
 - (a) $\mathcal{A}$ knows the value of Y_{k-1}^{*j} .
 - (b) $\mathcal{A}$ then sets $X_k^{*j} = (O_{\frac{n}{2}} \| \lfloor Y_{k-1}^{*j} \rfloor_{\frac{n}{2}}) \oplus C_{k-1}^{*j}$.
 - (c) It sends (tw_k^{*j}, X_k^{*j}) as a prediction query and receives Y_k^{*j} .

4.2.2 Understanding the reduction game

The adversary $\mathcal{A}$'s actions on receiving an encryption query are quite simple. To each decryption query, $\mathcal{A}$ simply responds $\perp$. Now, we try to understand how the adversary $\mathcal{A}$ generates the commitments and the predictions depending upon the queries of $\mathcal{B}$. Notice that for each decryption query by $\mathcal{B}$, $\mathcal{A}$ sets an integer flag p taking values in $[-1, m_i]$. Moreover, it makes at least one commitment and at least one prediction for each decryption query.

For simplicity, we assume that $\mathcal{B}$ makes only one decryption query of the form (N^*, A^*, C^*, T^*) .

Here, we only discuss the most complex case, i.e., when there exists an encryption query of the form (N, A, M) with response (C, T) , such that $N^* = N$; $l^* = l$ and $T^* = T$. The adversary looks for the maximum possible common suffix between C and C^* . Assume that the last p' blocks of C and C^* are identical. Then, depending on whether the most significant half of the last non-identical blocks of C and C^* are identical or not, the flag is set to $p' + 2$ or $p' + 1$, respectively. With the help of $\mathcal{CH}$, $\mathcal{A}$ simulates the mF decryption protocol to compute Y_{l-p+1}^* before exiting phase 1.

Note that adversary $\mathcal{A}$ knows all the $\{Y_{l-p+2}, \dots, Y_{l+1}\}$ values from the encryption transcript generated for $\mathcal{B}$. The adversary simply sets p commitments of the form $((N, l - p + 2), \lceil C_{m^* - p + 1} \rceil_{\frac{n}{2}}, \lfloor Y_{l-p+2} \rfloor_{\frac{n}{2}}), \dots, ((N, l + 1), \lceil C_{m^*} \rceil_{\frac{n}{2}}, \lfloor Y_{l+1} \rfloor_{\frac{n}{2}})$. Finally, $\mathcal{A}$ returns to simulating the decryption protocol starting from Y_{l-p+1}^* by sending prediction queries of the form $((N, l - p + k), X_{l-p+k}^*)_{k \in [2, p+1]}$ to $\mathcal{CH}$.

Remark 4.5. When $\mathcal{B}$ makes more than one decryption query, $\mathcal{A}$ doesn't make any prediction query before generating commitments corresponding to all the decryption queries.

Let CBAD denote the event that $\mathcal{A}$ receives an encryption query of the form (N^i, A^i, M^i) to output a response of the form (C^i, T^i) , such that, for some $1 \leq c \leq l_i$, we have $\lfloor X_c^i \rfloor_{\frac{n}{2}} = \lfloor Y_{c-1}^i \rfloor_{\frac{n}{2}} \oplus \lfloor \delta_M \rfloor_{\frac{n}{2}}$ for some arbitrary M .

Lemma 4.6.

$$\Pr[\text{CBAD}] \leq \frac{2\sigma_e}{2^{\frac{n}{2}}} + \text{Adv}_E^{\mu\text{-tprp}}(\mathcal{C}').$$

Proof. Since CBAD occurs only during the encryption queries, by a standard reduction technique, there exists a μ -TPRP adversary $\mathcal{C}'$ such that

$$\Pr[\text{CBAD}] \leq \Pr[\text{CBAD}_P] + \text{Adv}_E^{\mu\text{-tprp}}(\mathcal{C}'),$$

where CBAD_P denotes the event CBAD when $\mathcal{A}$ has oracle access to P .

Now, Let $\mathcal{A}$ have oracle access to P . Then, during the i th encryption query, we have

$$[X_c^i]_{\frac{n}{2}} = \begin{cases} [A_c^i]_{\frac{n}{2}} & \text{if } c \leq a_i \\ [Y_{c-1}^i]_{\frac{n}{2}} \oplus [\delta_{A^i}]_{\frac{n}{2}} & \text{if } c = a_i + 1 \\ [M_c^i]_{\frac{n}{2}} & \text{if } c > a_i + 1. \end{cases}$$

Now since all the Y_j^i values are generated uniformly at random and $[\delta_{A^i}]_{\frac{n}{2}} \neq [\delta_M]_{\frac{n}{2}}$ for any M . Hence, for any $i \in [q]$ and any M , $\Pr[\text{CBAD}] \leq \frac{1}{2^{\frac{n}{2}}}$. Since, δ_M takes at most two values, depending on whether $n|M$ or not, varying over all i, j , we have the lemma. $\square$

Corollary 4.7. *If the event CBAD doesn't hold, then for any decryption query $(N^{*j}, A^{*j}, C^{*j}, T^{*j})$, such that $N^{*j} = N^i$, $l_i > l_j^*$ and $T^{*j} = Y_{l_j^*+2}^i$, then $X_{l_j^*+1}^{*j} \neq X_{l_j^*+1}^i$.*

Proof. Note that, $T^{*j} = Y_{l_j^*+2}^i \Rightarrow X_{l_j^*+2}^{*j} = X_{l_j^*+2}^i$. Now, since $Y_{l_j^*+1}^{*j} = X_{l_j^*+2}^{*j} \oplus \delta_{M^{*j}}$ and since CBAD doesn't hold, $Y_{l_j^*+1}^{*j} \neq Y_{l_j^*+1}^i$, which implies $X_{l_j^*+1}^{*j} \neq X_{l_j^*+1}^i$. $\square$

Proposition 4.8. *Suppose $\mathcal{A}$ never Aborts and CBAD never occurs. If $(N^{*j}, A^{*j}, C^{*j}, T^{*j})$ is a valid forgery, for some $j \in [1, q_d]$, then for some $k \in [-1, p_j]$, we have $(\text{tw}_k^{*j}, X_k^{*j})$ is a successful prediction query tuple.*

We postpone the proof of Proposition 4.8 to Section 4.3.

4.2.3 Proof of Theorem 4.3

For all encryption queries of the form (N^i, A^i, M^i) , $\mathcal{A}$ can correctly simulate as it has access to $\tilde{E}_K$.

Note that Proposition 4.8 means, that, given $\mathcal{A}$ doesn't abort and CBAD doesn't occur for any encryption query by $\mathcal{B}$, the (μ, σ_d) -mcp adversary $\mathcal{A}$ makes a valid prediction whenever the forging adversary $\mathcal{B}$ makes a successful forgery. Hence, by Proposition 4.8,

$$\Pr[\mathcal{A} \text{ wins } (\mu, \sigma_d)\text{-mcp game}] \geq \Pr[\mathcal{B} \text{ Forges } i \text{ th query for some } i \in [1, q_d] | \mathcal{A} \text{ doesn't abort} \cap \overline{\text{CBAD}}]$$

Hence,

$$\begin{aligned} \Pr[\mathcal{B} \text{ Forges}] &\leq \Pr[\mathcal{B} \text{ Forges } i \text{ th query for some } i \in [1, q_d] | \mathcal{A} \text{ doesn't abort} \cap \overline{\text{CBAD}}] + \Pr[\mathcal{A} \text{ aborts}] \\ &\quad + \Pr[\text{CBAD}] \\ &\leq \Pr[\mathcal{A} \text{ wins } (\mu, \sigma_d)\text{-mcp game}] + \Pr[\mathcal{A} \text{ aborts}] + \Pr[\text{CBAD}] \\ &\leq \text{Adv}_{\tilde{E}}^{(\mu, \sigma_d)\text{-mcp}}(\mathcal{A}) + 2 \cdot \text{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(\mathcal{C}') + \sigma \left(1 + \frac{(\mu+1)^2}{2^n} \right) \left(\frac{\sigma}{2^{\frac{n}{2}}} \right)^{\mu} + \frac{2\sigma_e}{2^{\frac{n}{2}}}. \end{aligned}$$

4.3 Proof of Proposition 4.8

Let $(N^{*j}, A^{*j}, C^{*j}, T^{*j})$ be a valid forgery. Depending on the value of p_j , we divide it into three cases.

Case 1: If $p_j = -1$.

In the commitment phase, the adversary $\mathcal{A}$ commits $((N^{*j}, l_j^* + 2), [Y_{l_j^*+1}^{*j}]_{\frac{n}{2}}, [T^{*j}]_{\frac{n}{2}})$ as described earlier.

Notice that if $N^i \neq N^{*j}$ for all encryption query of the form (N^i, A^i, M^i) , then $(N^{*j}, l_j^* + 2)$ is fresh.

If $N^i = N^{*j}$ and $l_j^* = l_i$ but $T^i \neq T^{*j}$, then since $(N^{*j}, l_j^* + 2) = (N^i, l_i + 2)$, we must have $((N^{*j}, l_j^* + 2), X_{l_j^*+2}^{*j})$ is fresh.

If $N^i = N^{*j}$ and $l_j^* > l_i$, then we again have $(N^{*j}, l_j^* + 2)$ is fresh.

Let $N^i = N^{*j}$ and $l_j^* < l_i$. If $T^{*j} \neq Y_{l_j^*+2}^i$, then we have $((N^{*j}, l_j^* + 2), X_{l_j^*+2}^{*j})$ is fresh.

Hence, if any of the aforementioned condition is satisfied, then $((N^{*j}, l_j^* + 2), X_{l_j^*+2}^{*j})$ is fresh, i.e., $((N^{*j}, l_j^* + 2), X_{l_j^*+2}^{*j})$ has never been queried before by $\mathcal{A}$ to $\mathcal{CH}$, $\lceil X_{l_j^*+2}^{*j} \rceil_{\frac{n}{2}} = \lceil Y_{l_j^*+1}^{*j} \rceil_{\frac{n}{2}}$ and $\tilde{E}_K((N^{*j}, l_j^* + 2), X_{l_j^*+2}^{*j}) = T^{*j}$. Hence, we see that $((N^{*j}, l_j^* + 2), X_{l_j^*+2}^{*j})$ is a valid prediction query with respect to the commitment $((N^{*j}, l_j^* + 2), \lceil Y_{l_j^*+1}^{*j} \rceil_{\frac{n}{2}}, \lceil T^{*j} \rceil_{\frac{n}{2}})$.

Case 2: If $p_j = 0$

We have, $N^i = N^{*j}$, $l_j^* < l_i$ and $T^{*j} = Y_{l_j^*+2}^i$. Then, we must have $X_{l_j^*+2}^{*j} = X_{l_j^*+2}^i$ and

$$\lceil Y_{l_j^*+1}^{*j} \rceil_{\frac{n}{2}} = \lceil Y_{l_j^*+1}^i \rceil_{\frac{n}{2}} \oplus \lceil D_{l_j^*+1}^i \rceil_{\frac{n}{2}} \oplus \lceil \delta_{M^{*j}} \rceil_{\frac{n}{2}},$$

where $D_{l_j^*+1}^i$ is as defined in phase 2.

In the commitment phase, the adversary $\mathcal{A}$ commits $((N^{*j}, l_j^* + 1), \lceil C_{m_j^*}^{*j} \rceil_{\frac{n}{2}}, \lceil Y_{l_j^*+1}^{*j} \rceil_{\frac{n}{2}})$ as described earlier.

Now, by Proposition 4.7, we have $((N^{*j}, l_j^* + 1), X_{l_j^*+1}^{*j})$ is fresh. Moreover, it is a valid prediction query.

Case 3: If $p_j \neq -1, 0$

There exist an $i \in [1, q_e]$ such that $N^{*j} = N^i$, $a_j^* + m_j^* = a_i + m_i = l_j^*$, $T^{*j} = T^i$.

Now consider the two cases:

- (I) First, let $p_j' \in \mathbb{Z}_{\geq 0}$ be such that $C_{m_j^*-k}^{*j} = C_{m_i-k}^i, \forall k \in [0, p_j')$ and $C_{m_j^*-p_j'}^{*j} \neq C_{m_i-p_j'}^i$ but $\lceil C_{m_j^*-p_j'}^{*j} \rceil_{\frac{n}{2}} = \lceil C_{m_i-p_j'}^i \rceil_{\frac{n}{2}}$. In this case, $p_j = p_j' + 2$. We have, by suffix property, $\Delta^j \neq 0$ and

$$\lceil Y_{l_j^*-p_j+2}^{*j} \rceil_{\frac{n}{2}} = \lceil Y_{l_j^*-p_j+2}^i \rceil_{\frac{n}{2}} \oplus \Delta^j.$$

$$\text{i.e., } X_{l_j^*-p_j+2}^{*j} \neq X_{l_j^*-p_j+2}^i.$$

- (II) Now, let $p_j' \in \mathbb{Z}_{\geq 0}$ be such that $C_{m_j^*-k}^{*j} = C_{m_i-k}^i, \forall k \in [0, p_j)$ and $\lceil C_{m_j^*-p_j'}^{*j} \rceil_{\frac{n}{2}} \neq \lceil C_{m_i-p_j'}^i \rceil_{\frac{n}{2}}$. Then, $p_j = p_j' + 1$ and, by the suffix property,

$$\lceil Y_{l_j^*-p_j+2}^{*j} \rceil_{\frac{n}{2}} = \lceil Y_{l_j^*-p_j+2}^i \rceil_{\frac{n}{2}}.$$

Since $\lceil C_{m_j^*-p_j+1}^{*j} \rceil_{\frac{n}{2}} \neq \lceil C_{m_i-p_j+1}^i \rceil_{\frac{n}{2}}$,

$$X_{l_j^*-p_j+2}^{*j} \neq X_{l_j^*-p_j+2}^i.$$

Hence, we conclude that $(tw_{l_j^*-p_j+2}^{*j}, X_{l_j^*-p_j+2}^{*j})$ is fresh in both cases.

In the commitment phase, the adversary commits $(tw_k^{*j}, x_k^{*j}, y_k^{*j})$ for all $k \in [l_j^* - p_j + 2, l_j^* + 1]$.

If $(tw_{l_j^*-p_j+2}^{*j}, X_{l_j^*-p_j+2}^{*j})$ is a valid prediction query with respect to $(tw_{l_j^*-p_j+2}^{*j}, x_{l_j^*-p_j+2}^{*j}, y_{l_j^*-p_j+1}^{*j})$, we are done.

If not, then $C_{m_j^*-p_j+2}^{*j} = C_{m_i-p_j+2}^i$.

$$\lceil Y_{l_j^*-p_j+2}^{*j} \rceil_{\frac{n}{2}} \neq \lceil Y_{l_j^*-p_j+2}^i \rceil_{\frac{n}{2}}.$$

$$\text{i.e., } X_{l_j^*-p_j+3}^{*j} \neq X_{l_j^*-p_j+3}^i.$$

Hence, we have $(tw_{l_j^*-p_j+3}^{*j}, X_{l_j^*-p_j+3}^{*j})$ is fresh.

Inductively, suppose $(tw_{l_j^*}^{*j}, X_{l_j^*}^{*j})$ is not a valid prediction query. Then, as $C_{m_j^*}^{*j} = C_{m_i}^i$,

$$\lceil Y_{l_j^*}^{*j} \rceil_{\frac{n}{2}} \neq \lceil Y_{l_j^*}^i \rceil_{\frac{n}{2}},$$

$$\text{i.e., } X_{l_j^*+1}^{*j} \neq X_{l_j^*+1}^i.$$

Hence, $(tw_{l_j^*+1}^{*j}, X_{l_j^*+1}^{*j})$ is fresh.

Since $N^{*j} = N^i$, $a_j^* + m_j^* = a_i + m_i = l_j^*$, $T^{*j} = T^i \Rightarrow X_{l_j^*+2}^{*j} = X_{l_j^*+2}^i$. Hence,

$$Y_{l_j^*+1}^{*j} = Y_{l_j^*+1}^i.$$

Finally, since $(N^{*j}, A^{*j}, C^{*j}, T^{*j})$ is a valid forgery, it must be that

$$\tilde{E}_K(tw_{l_j^*+1}^{*j}, X_{l_j^*+1}^{*j}) = Y_{l_j^*+1}^{*j}.$$

Hence, $(tw_{l_j^*+1}^{*j}, X_{l_j^*+1}^{*j})$ must be a valid prediction query.

5 A block cipher-based TBC construction

Let $\rho : \{0, 1\}^n \rightarrow \{0, 1\}^n$ be any bijective function and ρ^i denotes i consecutive applications of ρ . We call ρ , the KUF of the TBC.

Definition 5.1. Given any fixed KUF ρ , define

$$v_\rho := \max_{l < 2^n} \frac{1}{l} \cdot \Pr \left[r_K \leq l \mid K \xleftarrow{\$} \{0, 1\}^n \right].$$

where for all $K \in \{0, 1\}^n$, r_K is defined as the smallest positive integer such that $\rho^{r_K}(K) = K$.

Notice that if $\rho(K) = \alpha \cdot K$, where α is a primitive polynomial of degree n , then $v_\rho = 0$. Leurent and Pernot [14] showed that if ρ is the 11th round-key function in the AES key scheduling algorithm, then $v_\rho \geq \frac{0.44}{14018661024}$.

Consider a block cipher $E : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^n$. Then, for any integer $t > n$, we define the TBC $\tilde{E} : \{0, 1\}^n \times \{0, 1\}^t \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ as, $\tilde{E}(K, tw, X) := E(K_{tw}, X)$, where $K_{tw} := \rho^i(E(K, N))$. Here, we parse tw to get $N = \lfloor tw \rfloor_n$; i is the decimal integer representation of $\lfloor tw \rfloor_{t-n}$ (Figure 6).

Remark 5.2. If the key size κ of the block cipher is less than the state size n , then we can take the ρ function with domain and range $\{0, 1\}^\kappa$ and chop K_N appropriately. If $\kappa > n$, we can generate the updated key suitably by multiple applications of ρ . Since there exist many popular block ciphers with $\kappa = n$, in this article we restrict our analysis to these types of block ciphers only.

5.1 Bounding μ -TPRP security of $\tilde{E}$

Here, we try to bound the μ -TPRP security of the TBC $\tilde{E}$. Let $\mathcal{A}$ be any μ -respecting adversary playing the μ -TPRP game that makes at most t primitive queries and d online queries.

We assume that the adversary doesn't make repetitive or redundant queries.

Figure 6: A block cipher-based TBC construction.

5.1.1 The ideal world and analysis of bad events

Let $\mathcal{P}$ and $\mathcal{E}$ denote the index set of primitive queries and encryption queries, respectively.

In the ideal world, the oracle chooses random functions $P : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ and $Q : T \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ such that for all $K \in \{0, 1\}^n$ we have $P(K, \star)$ as a random permutation and for all $tw \in T$, we have $Q(tw, \star)$ as a random permutation.

Primitive query: In the Ideal world for the i th primitive query of the form (K^i, X^i) , it computes $Y^i = P(K^i, X^i)$ and sends it as a response.

Define $\omega_t = (K^i, X^i, Y^i)_{i \in \mathcal{P}}$ to be the primitive transcript.

Online query: On receiving the i th input query of the form $((N^i, j^i), X^i)$, it computes $Y^i = Q((N^i, j^i), X^i)$ and sends it as the response.

Offline computation: Oracle chooses $K \in \{0, 1\}^n$ uniformly at random. It then chooses a permutation $\Pi : \{0, 1\}^n \rightarrow \{0, 1\}^n$ uniformly at random from the set of all permutations over $\{0, 1\}^n$. It then defines $K_{N^i} := \lfloor \Pi(N^i) \rfloor_n$ and $K^i = \rho^j(K_{N^i})$.

Define $\omega_d = (K, ((N^i, j^i), X^i, Y^i, K^i)_{i \in \mathcal{E}})$, to be the online transcript.

Define $\omega = (\omega_t, \omega_d)$ be the transcript for the adversary in the ideal world.

Bad Events: We now look at the ideal world transcript ω . We identify all the possible events where there is an input or output collision between different types of query-response tuples in ω , i.e., between the inputs or outputs of $(K^i, X_i, Y^i)_{i \in \mathcal{P}}$, $(K, tw^i, K^i)_{i \in \mathcal{E}}$ and $(K^i, X^i, Y^i)_{i \in \mathcal{E}}$. The six possible input collisions are as follows:

- (I1) $(K^i, X^i) = (K^{i'}, X^{i'})$, $i, i' \in \mathcal{P}$
- (I2) $(K, tw^i) = (K, tw^{i'})$, $i, i' \in \mathcal{E}$
- (I3) $(K^i, X^i) = (K^{i'}, X^{i'})$, $i, i' \in \mathcal{E}$
- (I4) $(K, tw^i) = (K^{i'}, X^{i'})$, $i \in \mathcal{E}, i' \in \mathcal{P}$
- (I5) $(K, tw^i) = (K^{i'}, X^{i'})$, $i, i' \in \mathcal{E}$
- (I6) $(K^i, X^i) = (K^{i'}, X^{i'})$, $i \in \mathcal{E}, i' \in \mathcal{P}$.

Similarly, the six possible output collisions are as follows:

- (O1) $(K^i, Y^i) = (K^{i'}, Y^{i'})$, $i, i' \in \mathcal{P}$
- (O2) $(K, K^i) = (K, K^{i'})$, $i, i' \in \mathcal{E}$
- (O3) $(K^i, Y^i) = (K^{i'}, Y^{i'})$, $i, i' \in \mathcal{E}$
- (O4) $(K, K^i) = (K^{i'}, Y^{i'})$, $i \in \mathcal{E}, i' \in \mathcal{P}$
- (O5) $(K, K^i) = (K^{i'}, Y^{i'})$, $i, i' \in \mathcal{E}$
- (O6) $(K^i, Y^i) = (K^{i'}, Y^{i'})$, $i \in \mathcal{E}, i' \in \mathcal{P}$.

We ignore cases I_1 and O_1 as the adversary doesn't make redundant queries. Similarly, we also ignore case I_2 as it simply means that the adversary has made multiple encryption queries with the same tweak. We consider the cases I_4, I_5, O_4 , and O_5 as subcases of the event that $K = K^i$ for some $i \in \mathcal{E} \cup \mathcal{P}$. We call this event as BAD₁. Similarly, we consider cases I_3, O_2 and O_3 as subcases of the event that for some $i, i' \in \mathcal{E}$ we have $K^i = K^{i'}$. Since the subcase where $tw^i = tw^{i'}$ is already considered in BAD₁, it is enough to consider the subcase, where $tw^i \neq tw^{i'}$. Define this event as BAD₂. We denote the event that case I_6 occurs as BAD₃. Finally, we denote the event that case O_6 occurs as BAD₄.

In notation:

BAD₁: For some $i \in \mathcal{E} \cup \mathcal{P}$, we have $K^i = K$.

BAD₂: For some $i_1 \neq i_2 \in \mathcal{E}$, we have $(N^{i_1}, j^{i_1}) \neq (N^{i_2}, j^{i_2})$ but $K^{i_1} = K^{i_2}$.

BAD₃: For some $i \in \mathcal{E}$ and $i' \in \mathcal{P}$, we have $(K^i, X^i) = (K^{i'}, X^{i'})$.

BAD₄: For some $i \in \mathcal{E}$ and $i' \in \mathcal{P}$, we have $(K^i, Y^i) = (K^{i'}, Y^{i'})$.

Note that, in the ideal world, BAD_1 implies that the adversary couldn't guess the secret key. Furthermore, BAD_2 , BAD_3 , and BAD_4 mean that the input–output tuples in ω_t and ω_d are distinct, i.e., permutation compatible.

Definition 5.3.

$$\text{BAD} = \bigcup_{i=1}^4 \text{BAD}_i.$$

We call a transcript ω bad if event BAD occurs.

Lemma 5.4.

$$\Pr[\text{BAD}] \leq d \cdot v_\rho + \frac{t+d}{2^n} + \frac{d^2}{2^{n+1}} + \frac{2\mu t}{2^n} + \frac{\binom{d}{\mu+1}}{(2^n)^\mu}.$$

Proof. Here, we try to bound the distinct bad events defined earlier.

Bounding BAD_1 : Fix $i \in \mathcal{P} \cup \mathcal{E}$. Since K is chosen uniformly at random, we have the probability that $K^i = K$ is at most $\frac{1}{2^n}$. similar varying over all i ,

$$\Pr[\text{BAD}_1] \leq \frac{d+t}{2^n}.$$

Bounding BAD_2 : This event can be divided into the following cases.

Case 1: ($N^{i_1} \neq N^{i_2}$) In this case, since Π is a random permutation, $K_{N^{i_1}} \neq K_{N^{i_2}}$ are distinct and independent. Hence, probability that $K^{i_1} = K^{i_2}$ is at most $\frac{1}{2^n}$. Varying over all $i_1, i_2 \in \mathcal{E}$, we have,

$$\Pr[\text{Case 1}] \leq \frac{d^2}{2^{n+1}}.$$

Case 2: ($N^{i_1} = N^{i_2}; j^{i_1} \neq j^{i_2}$) In this case, we have $K_{N^{i_1}} = K_{N^{i_2}}$.

Hence, Case 2 event occurs if and only if, $\rho^{(j^{i_1}-j^{i_2})}(K_{N^{i_1}}) = K_{N^{i_2}}$, i.e., $(j^{i_1} - j^{i_2})$ is divisible by the periodicity of $K_{N^{i_1}}$ (say $r_{K_{N^{i_1}}}$).

Note that queries of this form arise due to the encryption query of $\mathcal{B}$ with nonce N^i in the privacy game.

Let l_i denote the number of blocks in the encryption query of $\mathcal{B}$ with nonce N^i . Then for all i_1, i_2 , such that $N^{i_1} = N^{i_2} = N^i$, we have $|j^{i_1} - j^{i_2}| \leq l_i$, i.e., $r_i \leq l_i$, and by Definition 5.1, the probability that Case 2 holds is at most $l_i \cdot v_\rho$.

Now, varying over all possible i , and from the observation that $\sum_i l_i \leq d$, we have

$$\Pr[\text{Case 2}] \leq \sum_i l_i \cdot v_\rho \leq d \cdot v_\rho.$$

Since the aforementioned two cases are mutually exclusive, we have

$$\Pr[\text{BAD}_2] \leq \frac{d^2}{2^{n+1}} + d \cdot v_\rho.$$

Bounding BAD_3 : For a given $i' \in \mathcal{P}$, let the adversary make the primitive query $(K^{i'}, X^{i'})$. Then, there can be at most μ encryption query of the form $((N^{i_k}, j^{i_k}), X^{i'})_{k \in [1, \mu], i_k \in \mathcal{E}}$, and hence, at most μ $(K^{i_k}, X^{i'})_{k \in [1, \mu], i_k \in \mathcal{E}}$ tuples. Now, since K^{i_k} are chosen uniformly at random during encryption query, we have, for a given $i_k \in \mathcal{E}$, the probability that $K^{i_k} = K^{i'}$ is at most $\frac{1}{2^n}$. Hence, for a given $i' \in \mathcal{P}$, the probability that $\exists i \in \mathcal{E}$ s.t. $(K^i, X^i) = (K^{i'}, X^{i'})$ is at most $\frac{\mu}{2^n}$. Varying over all i' , we have

$$\Pr[\text{BAD}_3] \leq \frac{\mu t}{2^n}.$$

Bounding BAD_4 : To bound BAD_4 , we first define an event BADY as follows:

BADY : $\exists i_1, \dots, i_{\mu+1} \in \mathcal{E}$ s.t. $Y^{i_k} = Y^{i_l} \quad \forall k, l \in [1, \mu + 1]$.

Then, by union bound, we have

$$\Pr[\text{BAD}_4] \leq \Pr[\text{BADY}] + \Pr[\text{BAD}_4 | \overline{\text{BADY}}].$$

Bounding BADY : Since for each $i \in \mathcal{E}$, Y^i is chosen uniformly at random, given $i_1, \dots, i_{\mu+1} \in \mathcal{E}$, the probability that $Y^{i_j} = Y^{i_l}$, for all $j \in [1, \mu + 1]$ is at most $\frac{1}{(2^n)^\mu}$. Hence, varying over all choices of $i_1, \dots, i_{\mu+1}$, we have

$$\Pr[\text{BADY}] \leq \frac{\binom{d}{\mu+1}}{(2^n)^\mu}.$$

Bounding $\text{BAD}_4 | \overline{\text{BADY}}$: For a given $i' \in \mathcal{P}$, let the adversary's primitive transcript be $(K^{i'}, *, Y^{i'})$. Then, there can be at most μ encryption transcripts of the form $((N^{i_k}, j^{i_k}), *, Y^{i'})_{k \in [1, \mu], i_k \in \mathcal{E}}$, and hence, at most $\mu (K^{i_k}, Y^{i'})_{k \in [1, \mu], i_k \in \mathcal{E}}$ tuples. Since K^{i_k} are chosen uniformly at random during encryption query, we have for a given $i_k \in \mathcal{E}$, the probability that $K^{i_k} = K^{i'}$ is at most $\frac{1}{2^n}$. Hence, for a given $i' \in \mathcal{P}$ probability, that $\exists i \in \mathcal{E}$ s.t. $(K^i, Y^i) = (K^{i'}, Y^{i'})$ is at most $\frac{\mu}{2^n}$. Varying over all i' , we have

$$\Pr[\text{BAD}_4 | \overline{\text{BADY}}] \leq \frac{\mu t}{2^n}.$$

Hence, we get

$$\Pr[\text{BAD}_4] \leq \frac{\binom{d}{\mu+1}}{(2^n)^\mu} + \frac{\mu t}{2^n}.$$

Finally, adding all the probabilities, we get the lemma. $\square$

5.1.2 Real World and good transcript analysis

The real world has oracle E_K . All the primitive queries and the encryption queries are responded to based on the responses of E_K .

By good transcript, we mean any transcript, which is not bad. Now consider a good transcript $\omega = (\omega_t, \omega_d)$. Let Θ_0 and Θ_1 be the transcript random variable obtained in the ideal world and real world respectively.

Note that by definition of the good transcript, the input-outputs of ω_t and ω_d in the ideal world are independent yet permutation compatible. Hence, we have

$$\Pr[\Theta_0 = \omega] = \prod_{t_i} \frac{1}{(2^n)_{t_i}} \times \frac{1}{2^n} \times \frac{1}{(2^n)_d} \times \frac{1}{(2^n)_d},$$

where t_i denotes the number of primitive queries with the key $K_i' \in \{0, 1\}^n$, i.e., $\sum_i t_i = t$.

Now, note that in the real world, the primitive queries and online queries are permutation compatible.

Hence, we have $\Pr[\Theta_1 = \omega] = \prod_{k_i} \frac{1}{(2^n)_{k_i}} \times \frac{1}{2^n} \times \frac{1}{(2^n)_d}$, where $k_i = d_i + t_i$ such that t_i denotes the number of primitive queries with key K_i and d_i denotes the number of encryption queries of the form (N^l, j^l, X) such that $K^l = K_i$. Note that $\sum_i k_i = d + t$.

Hence,

$$\frac{\Pr[\Theta_1]}{\Pr[\Theta_0]} = \frac{\prod_{t_i}(2^n)_{t_i} \times 2^n \times (2^n)_d \times (2^n)_d}{\prod_{k_i}(2^n)_{k_i} \times 2^n \times (2^n)_d} = \frac{\prod_i(2^n)_{t_i} \times (2^n)_d}{\prod_i(2^n)_{t_i+d_i}} = \frac{(2^n)_d}{\prod_i(2^n - t_i)_{d_i}} > 1.$$

Hence, by H-coefficient technique, we have Theorem 5.5.

Theorem 5.5.

$$\text{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(d, t) \leq d \cdot \nu_\rho + \frac{t+d}{2^n} + \frac{d^2}{2^{n+1}} + \frac{2\mu t}{2^n} + \frac{\binom{d}{\mu+1}}{(2^n)^\mu}.$$

5.2 Bounding (μ, λ) -mcp security of $\tilde{E}$

Here, we try to bound the advantage of a μ -respecting adversary $\mathcal{A}$ making t primitive queries and d online queries playing the (μ, λ) -multi commitment prediction game with a challenger $\mathcal{CH}$. We assume that the adversary doesn't make repetitive or redundant queries.

5.2.1 Game 0:

We define the original (μ, λ) -mcp security game between $\mathcal{A}$ and $\mathcal{CH}$ as Game 0. Define $\mathcal{P}$, $\mathcal{E}_1$, and $\mathcal{E}_2$, respectively as the set of query indices of Phase 1 primitive queries, Phase 1 encryption queries, and Phase 2 encryption queries.

Phase 1:

Primitive query: For the i th primitive query of the form $(K^i, X^i)_{i \in \mathcal{P}}$, $\mathcal{CH}$ computes $Y^i = E(K^i, X^i)$ and sends it as a response.

Define $\omega_t = (K^i, X^i, Y^i)_{i \in \mathcal{P}}$ to be the primitive transcript.

Online query oracle chooses $K \in \{0, 1\}^n$ uniformly at random. On receiving the i th input query of the form $((N^i, j^i), X^i)_{i \in \mathcal{E}_1}$ if the query is μ -respecting, then $\mathcal{CH}$ computes $K_{N^i} = E(K, N^i)$, $K^i = \rho^{j^i}(K_{N^i})$ and outputs $Y^i = E(K^i, X^i)$ as response. Else, it aborts.

Phase 2:

Commitment generation: $\mathcal{A}$ sends λ commitments of the form $(tw^i, x^i, y^i)_{i \in [1, \lambda]}$ to $\mathcal{CH}$.

Primitive queries: $\mathcal{A}$ doesn't make any primitive query in phase 2.

Prediction queries: Whenever $\mathcal{A}$ makes a fresh prediction query of the form $((N^i, j^i), X^i)$ for some $i \in \mathcal{E}_2$, $\mathcal{CH}$ computes $K_{N^i} = E(K, N^i)$, $K^i = \rho^{j^i}(K_{N^i})$ and outputs $Y^i = E(K^i, X^i)$ as response.

Let $\omega_{e_1} = ((N^i, j^i), X^i, Y^i)_{i \in \mathcal{E}_1}$ and $\omega_{e_2} = ((N^i, j^i), X^i, Y^i)_{i \in \mathcal{E}_2}$ be the phases 1 and 2, respectively, online transcript of the adversary.

Define $\omega_e = \omega_{e_1} \cup \omega_{e_2}$ as the overall online transcript of the adversary. Define $\omega = (\omega_t, \omega_e)$ as the transcript of $\mathcal{A}$.

5.2.2 Game 1:

We now define a newly modified security game called Game 1. Here, $\mathcal{CH}$ chooses random a function $Q : T \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ such that for all $tw \in T$ we have $Q(tw, *)$ is a random permutation. $\mathcal{CH}$ acts similar to a Game 0 challenger except in the case of phase 1 online queries.

Phase 1 online query: On receiving the i th input query of the form $((N^i, j^i), X^i)_{i \in \mathcal{E}_1}$ if the query is μ -respecting, then $\mathcal{CH}$ computes $Y^i = Q((N^i, j^i), X^i)$ and sends it as the response. Else, it aborts.

We say that any adversary $\mathcal{A}$ wins Game 1 if for some prediction query tuple $((N^k, j^k), X^k)$ there exist a commitment tuple (tw^i, x^i, y^i) such that

$$tw_i = (N^k, j^k); x_i = [X^k]_{\frac{n}{2}}; [\tilde{E}_K(tw^i, X^k)]_{\frac{n}{2}} = y^i.$$

$$\mathbf{Adv}_E^{\text{Game 1}}(d, t) = \max_{\mathcal{A}} \Pr[\mathcal{A}^{\tilde{E}} \text{ wins Game 1}].$$

where the maximum is taken over all adversaries $\mathcal{A}$ running in time t making at most d queries.

Proposition 5.6. *Given any (d, t) -adversary $\mathcal{A}$ playing Game 0 (or Game 1), there exists a $(d, t + 2d)$ -adversary $\mathcal{B}$ playing μ -TPRP security game such that,*

$$\Pr[\mathcal{A} \text{ wins Game 0}] \leq \Pr[\mathcal{A} \text{ wins Game 1}] + \mathbf{Adv}_E^{\mu\text{-tprp}}(\mathcal{B}).$$

Proof. We construct the $(d, t + 2d)$ -adversary $\mathcal{B}$ playing against an a μ -TPRP challenger $\mathcal{CH}$ as follows:

1. Whenever $\mathcal{A}$ makes a phase 1 primitive query $\mathcal{B}$ makes the same primitive query to the μ -TPRP challenger $\mathcal{CH}$ and forward the response to $\mathcal{A}$.
2. $\mathcal{B}$ chooses $K \in \{0, 1\}^n$ uniformly at random.
3. Whenever $\mathcal{A}$ makes a phase 1 encryption query $\mathcal{B}$ makes the same encryption query to the μ -TPRP challenger $\mathcal{CH}$ and forward the response to $\mathcal{A}$.
4. On receiving the commitments from $\mathcal{A}$, $\mathcal{B}$ does nothing.
5. Whenever $\mathcal{A}$ makes a prediction query of the form $((N^i, j^i), X^i)$ for some $i \in \mathcal{E}_2$, $\mathcal{B}$ makes a primitive query to $\mathcal{CH}$ to receive $K_{N^i} = E(K, N^i)$. It then computes $K^i = \rho^{j^i}(K_{N^i})$ and makes a second primitive query to output $Y^i = E(K^i, X)$ as response.
6. Whenever $\mathcal{A}$ wins (resp. loses) $\mathcal{B}$ sends 1 (resp. 0) to $\mathcal{CH}$.

From the construction, it is clear that whenever $\mathcal{CH}$ is a real (resp. ideal) oracle adversary, $\mathcal{B}$ simulates perfectly as a Game 0 (resp. Game 1) oracle to the adversary $\mathcal{A}$. Hence,

$$\begin{aligned} \Pr[\mathcal{A} \text{ wins Game 0}] &= \Pr[\mathcal{B} \rightarrow 1 \mid \mathcal{CH} \text{ real}] \\ \Pr[\mathcal{A} \text{ wins Game 1}] &= \Pr[\mathcal{B} \rightarrow 1 \mid \mathcal{CH} \text{ ideal}]. \end{aligned}$$

Hence,

$$\begin{aligned} |\Pr[\mathcal{A} \text{ wins Game 0}] - \Pr[\mathcal{A} \text{ wins Game 1}]| &= |\Pr[\mathcal{B} \rightarrow 1 \mid \mathcal{CH} \text{ real}] - \Pr[\mathcal{B} \rightarrow 1 \mid \mathcal{CH} \text{ ideal}]| \\ &= \mathbf{Adv}_E^{\mu\text{-tprp}}(\mathcal{B}) \end{aligned} \quad \square$$

Proposition 5.7.

$$\mathbf{Adv}_E^{\text{Game 1}}(d, t) \leq \frac{\lambda t}{2^{\frac{3n}{2}}} + \frac{\lambda}{2^{\frac{n}{2}-1}}.$$

Proof. Consider the following event due to Phase 2.

BAD₅: For some $i \in [1, \lambda]$ and $i' \in \mathcal{P}$, we have a commitment $((N^i, j^i), x^i, y^i)$ such that, $(K^i, x^i) = (K^{i'}, [X^{i'}]_{\frac{n}{2}})$, where $K^i := \rho^{j^i} \cdot E_K(N^i)$. $\square$

Claim 5.8.

$$\Pr[\text{BAD}_5] \leq \frac{\lambda t}{2^{\frac{3n}{2}}}.$$

Proof. Fix $i \in [1, \lambda]$ and $i' \in \mathcal{P}$. Since K_{N^i} is distributed uniformly at random, and there is no primitive query after commitment, we have the probability that $(K^i, x^i) = (K^{i'}, \lceil X^{i'} \rceil_{\frac{n}{2}})$ is at most $\frac{1}{2^{\frac{3n}{2}}}$. Varying over all i, i' , we have the claim. $\square$

Claim 5.9.

$$\Pr[\mathcal{A} \text{ wins Game 1} \mid \overline{\text{BAD}_5}] \leq \frac{\lambda}{2^{\frac{n}{2}-1}}.$$

Proof. Suppose $((N^i, j^i), X^i)$ is a valid prediction for some $i \in \mathcal{E}_2$. Let (tw^j, x^j, y^j) be the commitment corresponding to this prediction. Since BAD_5 doesn't occur, there is no primitive query of the form (K^i, X^i) in phase 1. Now suppose there are κ_i many primitive queries in phase 1 of the form $(K^i, \star)$. Then, the probability that $\lfloor Y^i \rfloor_{\frac{n}{2}} = y_i$ is bounded by $\frac{2^{\frac{n}{2}}}{2^n - \kappa}$. Since $\kappa \leq t$, assuming $t \leq 2^{n-1}$ and varying over all i , we have the claim. $\square$

Proposition 5.7 follows from Claims 5.8 and 5.9.

Theorem 5.10.

$$\text{Adv}_{\tilde{E}}^{(\mu, \lambda)\text{-mcp}}(d, t) \leq d \cdot v_\rho + \frac{t + 3d}{2^n} + \frac{d^2}{2^{n+1}} + \frac{2\mu(t + 2d)}{2^n} + \frac{\binom{d}{\mu+1}}{(2^n)^\mu} + \frac{\lambda t}{2^{\frac{3n}{2}}} + \frac{\lambda}{2^{\frac{n}{2}-1}}.$$

Proof. From Proposition 5.6, we have for any (d, t) adversary $\mathcal{A}$ we have we have a $(d, t + 2d)$ adversary $\mathcal{B}$ such that

$$\text{Adv}^{\text{Game 0}}(\mathcal{A}) \leq \text{Adv}^{\text{Game 1}}(\mathcal{A}) + \text{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(\mathcal{B}).$$

Taking maximum over all such (d, t) adversaries $\mathcal{A}$, we have

$$\text{Adv}^{\text{Game 0}}(d, t) \leq \text{Adv}^{\text{Game 1}}(d, t) + \text{Adv}_{\tilde{E}}^{\mu\text{-tprp}}(d, t + 2d).$$

Now plugging in the appropriate values from Proposition 5.7 and Theorem 4.1, we have Theorem 4.3. $\square$

6 mF under the new TBC construction

In this section, we consider the mF construction under the new TBC construction defined in Section 5.

Theorem 6.1.

$$\text{Adv}_{\text{mF}, \tilde{E}}^{\text{priv}}(\sigma, t) \leq \sigma v_\rho + \frac{t + \sigma}{2^n} + \frac{\sigma^2}{2^{n+1}} + \frac{2\mu t}{2^n} + \frac{\binom{\sigma}{\mu+1}}{(2^n)^\mu} + \sigma \left(1 + \frac{(\mu + 1)^2}{2^n} \right) \left(\frac{\sigma}{2^{\frac{n}{2}}} \right)^\mu.$$

Theorem 6.2.

$$\begin{aligned} \text{Adv}_{\text{mF}, \tilde{E}}^{\text{forge}}(\sigma, t) &\leq 3\sigma \cdot v_\rho + \frac{3(1 + 2\mu)t}{2^n} + \frac{3\sigma^2}{2^{n+1}} + \frac{(5 + 4\mu)\sigma}{2^n} + \frac{3\binom{\sigma}{\mu+1}}{(2^n)^\mu} \\ &\quad + \frac{\sigma t}{2^{\frac{3n}{2}}} + \frac{\sigma}{2^{\frac{n}{2}-2}} + \sigma \left(1 + \frac{(\mu + 1)^2}{2^n} \right) \left(\frac{\sigma}{2^{\frac{n}{2}}} \right)^\mu, \end{aligned}$$

where n is the state size and the key size and μ is the number of multi collisions allowed in the input of the TBC. For all calculation purposes, take $\mu \geq 4$.

Proof. Theorems 6.1 and 6.2 can be derived from Theorems 4.1 and 4.3, respectively, by appropriately plugging in the security bounds for $\tilde{E}$ derived in Section 5. $\square$

6.1 mixFeed as an mF construction

The mixFeed [6] mode of AEAD uses a block cipher AES', which is a variation of AES [2] in the sense that, unlike the original scheme, it calls the AES-MixColumn operation in the last round and also outputs the 11th round key in the AES key schedule. It then uses the new key to process the next data block. Since the key outputs only depend on the previous key input and are independent of the data inputs, this operation can be run in parallel and mixFeed can be viewed as an mF construction with $n = 128$ and the 11th round key function in the AES key scheduling algorithm as the KUF.

Khairallah [13] observed that mixFeed is prone to practical forgery due to the existence of small periodic cycles in AES key schedule algorithm. Later, Leurent and Pernot [14] confirmed Mustafa's observations by giving a practical attack on mixFeed with a success probability of 0.44 (with data complexity 220GB). In our notations, if $l = 14018661024$, then $\Pr\left[r_K \leq l, K \xleftarrow{\$} \{0, 1\}^{128}\right] \approx 0.44$. Plugging in this value in Definition 5.1, we get, $\sigma v_p \geq \frac{2^{46} \times 0.44}{14018661024} \gg 1$ in Theorems 6.1 and 6.2.

7 Overcoming the weakness of mixFeed

In this section, we show that the weakness in mixFeed [6] is only a weakness of the AES [2] key schedule and not of the AEAD mode in general. More specifically, we describe a specific mF construction and show that it is well secured within the NIST prescribed bounds.

For any primitive polynomial α of degree n , we define $\rho(x) = \alpha \cdot x \ \forall x \in \{0, 1\}^n$. Consider the TBC construction of Section 6 with ρ as its KUF.

Next, we consider the mF mode of AEAD with this TBC and call it mF_{prim} . As noted earlier, $v_p = 0$. Hence, from Theorems 6.1 and 6.2, we can conclude that for any adversary running in time t and making at most q encryption and decryption (in case of forgery) query with a total of at most σ blocks,

Theorem 7.1.

$$\text{Adv}_{\text{mF}_{\text{prim}}}^{\text{priv}}(\sigma, t) \leq \frac{t + \sigma}{2^n} + \frac{\sigma^2}{2^{n+1}} + \frac{2\mu t}{2^n} + \frac{\binom{\sigma}{\mu+1}}{(2^n)^\mu} + \sigma \left(1 + \frac{(\mu+1)^2}{2^n}\right) \left(\frac{\sigma}{2^{\frac{n}{2}}}\right)^\mu.$$

Theorem 7.2.

$$\begin{aligned} \text{Adv}_{\text{mF}_{\text{prim}}, \tilde{E}}^{\text{forge}}(\sigma, t) \leq & \frac{3(1+2\mu)t}{2^n} + \frac{3\sigma^2}{2^{n+1}} + \frac{(5+4\mu)\sigma}{2^n} + \frac{3\binom{\sigma}{\mu+1}}{(2^n)^\mu} \\ & + \frac{\sigma t}{2^{\frac{3n}{2}}} + \frac{\sigma}{2^{\frac{n}{2}-2}} + \sigma \left(1 + \frac{(\mu+1)^2}{2^n}\right) \left(\frac{\sigma}{2^{\frac{n}{2}}}\right)^\mu, \end{aligned}$$

Figure 7: A TBC in mF with linear KUF. Here, α is any primitive polynomial of degree n .

where n is the state size and μ is the number of multi collisions allowed in the input of the TBC. For all calculations, take $\mu \geq 4$.

7.1 Interpretation of the above bounds

According to NIST requirement, $\sigma \leq 2^{46}$ and $t \leq 2^{112}$. Following the recommendation in ref. [6], we take $n = \kappa = 128$. Then, taking $\mu = 4$, we have $\sigma \left(1 + \frac{(\mu+1)^2}{2^n}\right) \left(\frac{\sigma}{2^{\frac{n}{2}}}\right)^\mu < 2^{-25}$, and hence, the dominating term is $\frac{2\mu t}{2^n}$ in Theorem 7.1 and $\frac{5\mu t}{2^n} + \frac{3\sigma}{2^{\frac{n}{2}}}$ in Theorem 7.2, which are both less than 2^{-10} . Hence, we conclude that the mF_{prim} mode is well secured within the complexity bounds specification of NIST.

Remark 7.3. If the linearity of multiplication by a primitive polynomial α becomes problematic under some block cipher designs, one could define $\rho(K) = P^{-1}(\alpha \cdot P(K))$ for some nonlinear permutation P . This preserves the cycle structure of the multiplication by α but can be arbitrarily nonlinear depending on P .

8 mF mode as a lightweight AEAD

In this section, we try to give a theoretical comparison between the mF_{prim} mode and some other TBC designs in the NIST LWC [4] competition. Ignoring the different types of overheads required in practical implementations, we define the state size as the number of bits required to hold the key, auxiliary keys such as masking key (if any), block cipher state, and round key. Note that the mF_{prim} mode of AEAD as in Section 7 can be implemented as a block cipher-based construction. Unlike Figure 7, where each TBC call can be seen to be using two block cipher calls, we can process the nonce N with key K only once to get K_N , and then we store K_N as the initial key. The rekeying can be done in parallel, by applying the KUF (ρ) to the previous key while processing each data block. In this way, we can process the whole encryption–decryption query with only one extra block cipher call. Further, the number of bits processed per primitive block cipher call is asymptotical to n .

Table 1: A theoretical comparison of different TBC-based lightweight AEAD schemes. Here the TBC of mF_{prim} is considered with AES-128/128 as the underlying block cipher

Mode	State size (includes key size)	Block size	Tweak size	# Pass	Bits processed per primitive call	Inverse free
Romulus-N1 [8]	512	128	384	1	128	Yes
Romulus- M1 [8]	512	128	384	2	64	Yes
SKINNY-AEAD (M1) [16]	640	128	384	1	128	No
QAMELEON [17]	640	128	384	1	128	No
LILLIPUT-I-128 [18]	576	128	320	1	128	No
mF_{prim}	384	128	NA	1	128	Yes

We have tabulated theoretical comparisons of different TBC-based AEAD schemes in Table 1. A more practical, implementation-based comparison is beyond the scope of this article and can be left as a future research problem.

9 Conclusion

In this article, we have considered a TBC-based AEAD scheme mF , which can be viewed as an abstraction of $mixFeed$ mode. We have proven that the security of the said mF mode can be reduced to the security of its underlying TBC. We constructed a new block cipher-based TBC construction and bound the security advantages of any adversary against mF mode using this TBC. We have tried to interpret the results of ref. [14] in our notations and confirmed the observation made in ref. [13] that in the case of $mixFeed$ the security of the underlying TBC depends on the periodicity of the AES key scheduling algorithm. Finally, to show that the said weakness is restricted to the use of AES key scheduling algorithm and that it doesn't affect the mF mode in general, we have constructed an explicit TBC construction and showed that the mF_{prim} mode using this TBC achieves the desired security within the NIST parameters.

Conflict of interest: Authors state no conflict of interest.

References

- [1] N. Mouha. The design space of lightweight cryptography. in: NIST Lightweight Cryptography Workshop 2015. 2015.
- [2] NIST. Announcing the ADVANCED ENCRYPTION STANDARD (AES), National Institute of Standards and Technology, U.S. Department of Commerce, Federal Information Processing Standards Publication no FIPS 197. 2001.
- [3] M. J. Dworkin. SHA-3 standard: Permutation-based hash and extendable-output functions. 2015.
- [4] NIST. Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process. 2018. <https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf>
- [5] M. Liskov, R.L. Rivest, D. Wagner. Tweakable block ciphers. In: Annual International Cryptology Conference. Springer; 2002. p. 31–46.
- [6] B. Chakraborty, M. Nandi. $mixFeed$. Submission to NIST LwC Standardization Process (Round 2). 2019. <https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/mixFeed-spec-round2.pdf>.
- [7] T. Iwata, M. Khairallah, K. Minematsu, T. Peyrin. REMUS. Submission to NIST LwC Standardization Process (Round 1). 2019. <https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/Remus-spec.pdf>.
- [8] T. Iwata, M. Khairallah, K. Minematsu, T. Peyrin. Romulus. Submission to NIST LwC Standardization Process (Round 2). 2019. <https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/Romulus-spec-round2.pdf>.
- [9] T. Iwata, M. Khairallah, K. Minematsu, T. Peyrin, Y. Sasaki, S. MengSim, L. Sun. Thank Goodness It's Friday (TGIF). Submission to NIST LwC Standardization Process (Round 1). 2019. <https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/TGIF-spec.pdf>.
- [10] S. Gueron, A. Jha, M. Nandi. COMET. Submission to NIST LwC Standardization Process (Round 1). 2019. <https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/comet-spec-round2.pdf>.
- [11] P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer; 2004. p. 16–31.
- [12] N. Datta, A. Jha, A. Mège, M. Nandi. Breaking REMUS and TGIF in the light of NIST Lightweight Cryptography Standardization Project. 2019. <https://csrc.nist.gov/CSRC/media/Events/lightweight-cryptography-workshop-2019/documents/papers/breaking-remus-and-tgif-lwc2019.pdf>.
- [13] M. Khairallah. Weak Keys in the Rekeying Paradigm: Application to COMET and $mixFeed$. Cryptology ePrint Archive, Report 2019/888. 2019. <https://eprint.iacr.org/2019/888>.
- [14] G. Leurent, C. Pernot. New Representations of the AES Key Schedule. Cryptology ePrint Archive, Report 2020/1253. 2020. <https://eprint.iacr.org/2020/1253>.

- [15] P. Rogaway. Authenticated-encryption with associated-data. In: Proceedings of the 9th ACM Conference on Computer and communications Security; 2002. p. 98–107.
- [16] Beierle C, Jean J, Kölbl S, Leander G, Moradi A, Peyrin T, et al. SKINNY-AEAD and SKINNY-HASH. Submission to NIST LwC Standardization Process (Round 2). 2019. <https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/SKINNY-spec-round2.pdf>.
- [17] R. Avanzi, S. Banik, A. Bogdanov, O. Dunkelman, S. Huang, F. Regazzoni. Qameleon v. 1.0. Submission to NIST LwC Standardization Process (Round 1). 2019. <https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/qameleon-spec.pdf>.
- [18] Adomnicai A, Berger TP, Clavier C, Francq J, Huynh P, Lallemand V, et al. Lilliput-AE: a new lightweight tweakable block cipher for authenticated encryption with associated data. Submitted to NIST Lightweight Project. 2019.