Misuse Resistant Parallel Authenticated Encryptions

Nilanjan Datta and Mridul Nandi

Cryptology Research Group
Applied Statistics Unit
Indian Statistical Institute
203, B.T. Road, Kolkata, India 700108
nilanjan_isi_jrf@yahoo.com, mridul.nandi@gmail.com

Abstract. The authenticated encryptions which resist misuse of initial value (or nonce) at some desired level of privacy are two-pass or Mac-then-Encrypt constructions (inherently inefficient but provide full privacy) and online constructions, e.g., McOE, sponge-type authenticated encryptions (such as duplex, AEGIS) and COPA. Only the last one is almost parallelizable with some bottleneck in processing associated data. In this paper, *we design a new online secure authenticated encryption, called* ELmE *or Encrypt-Linear mix-Encrypt, which is completely (two-stage)* **parallel** *(even in associated data) and* **pipeline implementable**. It also provides full privacy when associated data (which includes initial value) is not repeated. The basic idea of our construction and COPA are based on EME, an Encrypt-Mix-Encrypt type SPRP constructions (secure against chosen plaintext and ciphertext). Unlike EME, we consider (so does COPA) online computable **linear mixing**. In addition with getting rid of bottleneck, our construction optionally supports **intermediate tags** which can be verified faster with less buffer size. Intermediate tag provides security against block-wise adversaries which is meaningful in low-end device implementation.

Keywords: Authenticated Encryption, Privacy, Misuse Resistant, EME.

1 Introduction

The common application of cryptography is to implement a secure channel between two or more users and then exchanging information over that channel. These users can initially set up their one-time shared key. Otherwise, a typical implementation first calls a key-exchange protocol for establishing a shared key or a session key (used only for the current session). Once the users have a shared key, either through the initial key set-up or key-exchange, they use this key to authenticate and encrypt the transmitted information using efficient symmetric-key algorithms such as a *message authentication code* $\mathsf{Mac}(\cdot)$ and (symmetric-key) *encryption* $\mathsf{Enc}(\cdot)$. The encryption provides **privacy** or **confidentiality** (hiding the sensitive data M, we call it *plaintext* or *message*) resulting a ciphertext C, whereas a message authentication code provides **data-integrity** (authenticating the transmitted message M or the ciphertext C) resulting a tag T. An authenticated encryption or AE is an integrated scheme which provides both privacy of plaintext and authenticity or data integrity of message or ciphertext. An authenticated encryption scheme F_K takes associated data D (which may include initial value or nonce) and message M and produces tagged-ciphertext (C, T). Its inverse F_K^{-1} returns $\perp$ for all those (D, C, T) for which no such M exists, otherwise it returns M. Note that the associated data D must be sent along with tagged-ciphertext to decrypt correctly. In case of IV (or nonce) based authenticated encryption schemes [33, 7], the IV must be distinct for every invocation of the tagged-encryption. Failure to do so, leads several critical attacks on the schemes. Usually, we apply a counter or we choose it randomly (then repetition can happen with negligible probability) to ensure distinct IV have been used in tagged-encryption. In this paper we do not need to have distinct IV and it still provides some amount of privacy, called **online privacy**.

1.1 Examples of Authenticated Encryptions

So far, cryptography community put a lot of effort of designing different authenticated encryptions. Lack of being standardized of this notion so far motivates to make a call for CEASER standard of AE [1, 2, 3]. We believe that our proposed construction would be a strong candidate for this competition. We quickly mention some of the popularly known competitive constructions putting into different categories based on construction types.

Encrypt-and-MAC and Encrypt-then-MAC. It relies on non-repeating IV (or nonce), e.g. CCM [17], EAX [6], GCM [37], CHM [18], CWC [23], Sarkar's generic construction [36] and dedicated Stream Ciphers like Grain [16], Helix [12], Zuc [4] etc. All these constructions combines counter type encryption and a Mac.

MAC-then-Encrypt. It is a two-pass IV misuse resistant category e.g., SIV [35], BTM [20], HBS [19]. These compute a tag first and then based on this tag, counter type encryption is used to encrypt.

Online Feed Back Encryption. It uses feedback type encryption, e.g. IACBC [22], XCBC [9], CCFB [26], McOE [13], sponge-type constructions (Duplex [8], AEGIS [30, 3] etc.). The construction McOE [13] (uses MHCBC [27], later generalized and called TC3 [34]) has a hardware bottleneck of not being fully pipelined (see the bottom layer of McOE in Figure 1.1. All these constructions are not fully parallelizable. Our construction ELmE and COPA [5] fall in this category which use basic structure of completely parallel EME, Encrypt-Mix-Encrypt constructions [15] with linear mixing in the middle layer.

Encrypt-then-Checksum. It uses IV-based block-wise encryption (non-repeating IV is required) and then finally checksum is used to compute tag. For example, different versions of OCB [7, 32, 24] and IAPM [22].

Quick comparison of our construction ELmE with COPA

Parallel in Both Message and Associated Data. Both, COPA and ELmE

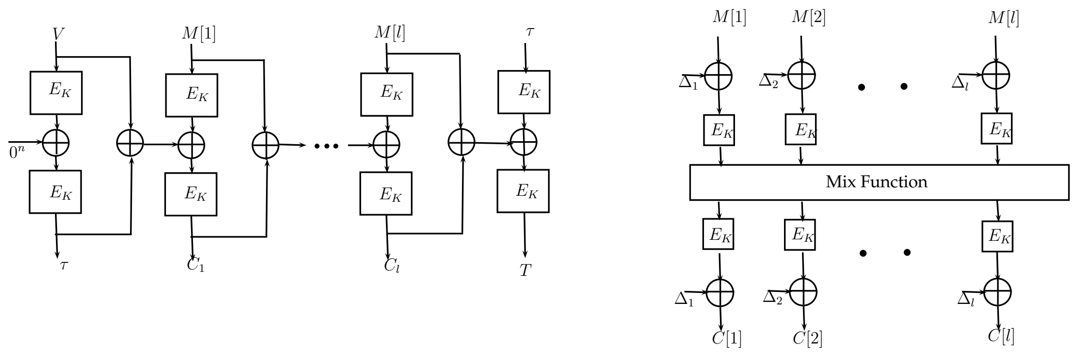

Fig. 1.1. (1) McOE-D construction : cannot be pipelined. (2) Encrypt-Mix-Encrypt : completely parallel and pipeline implementable.

require about two calls of blockcipher for every message block. However, COPA has a bottleneck in processing associated data. In our construction we process associate data and message in a similar fashion and these are completely parallel.
Area in Combined Implementation. Implementation of encryption and decryption requires two separate layers of encryption and decryption (some of the architectures however can be shared). Our construction replaces the second layer encryption by decryption which makes authenticated encryption and verified decryption almost identical. It requires one layer of encryption and decryption.

Intermediate Tag: A Solution against Block-wise Adaptive Adversaries. Finally, we show how to efficiently define intermediate tag. The intermediate tag is useful when we implement it in a limited buffer environment and hence it become exposed against block-wise adaptive adversaries. Unlike McOE, the COPA (as well as our construction without intermediate tag) is vulnerable. Verifying integrity of the part of the message so far processed can resist against these attacks. Even though designers of COPA did not propose any intermediate tag, a natural way to obtain intermediate tag for COPA is to compute tag (as if the the final tag) for the part of the message. Due to choice of our mixing layers, tag verification can be done much faster than the natural choice of intermediate tag of COPA and it requires about half size buffer than that of COPA. Our method of tag generation would not be applicable to COPA due to the choice of their linear mixing.

Outline of the paper. In this paper we have proposed ELmE, an online authenticated encryption, which is fully parallelizable and pipelined as we have described before. After providing basic preliminaries in section 2, we state and illustrate our construction in section 3. In section 4 and 5, we prove the privacy and authenticity. In section 6, we show how an intermediate tag can be optionally adjoined to our construction. The same method does not work for COPA and the possible intermediate tag would require more buffer size to implement (it is important in those low-end environments where intermediate tags are required) that of our. We also sketch the privacy and authenticity of the construction with intermediate tag. Finally we conclude along with some possible future works.

2 Preliminaries

Definitions and Notation. By convention, $\mathbb{B} = \{0,1\}^n$ where n is the block size of the underlying blockcipher. An ℓ-tuple $x \in \mathbb{B}^\ell$ is denoted by $(x[1], x[2], \ldots, x[\ell])$. We call $\ell := \|x\|$ block-length of x. For $0 \le a \le b < \ell$ we denote $x[a..b] := (x[a],\ x[a+1], \ldots,\ x[b])$, $x[..b] = x[1..b]$. Let us fix q message and associate data pairs $P_1 = (D_1, M_1), \ldots, P_q = (D_q, M_q)$ with $\|D_i\| = d_i$, $\|M_i\| = e_i$, $\ell_i = d_i + e_i$. We denote $(P_1, \ldots, P_q)$ by τ_{in}. We assume that all P_i's are distinct and in case D_i contains distinct initial value, we call it nonce-respecting. A tagged ciphertext tuple $\tau_{out} = (C_1, T_1, \ldots, C_q, T_q)$ (also the complete view $\tau = (\tau_{in}, \tau_{out})$) is called **online view** if for all i, $\|C_i\| = e_i$ and $C_i[..j] = C_{i'}[..j]$ whenever $D_i = D_{i'}$ and $M_i[..j] = M_{i'}[..j]$.

2.1 Full and Online Privacy

We give a particularly strong definition of privacy, one asserting indistinguishability from random strings. Consider an adversary A who has access of one of two types of oracles: a "real" encryption oracle or an "ideal" authenticated encryption oracle. A real authenticated encryption oracle, F_K, takes as input (D, M) and returns $(C, T) = F_K(D, M)$. Whereas an ideal authenticated encryption oracle \$ returns a random string R with $\|R\| = \|M\| + 1$ for every fresh pair (D, M). Given an adversary A (w.o.l.g. throughout the paper we assume a **deterministic adversary**) and an authenticated encryption scheme F, we define the (full) **privacy-advantage** of A by the distinguishing advantage of A distinguishing F from \$. More formally,

$$\mathbf{Adv}_F^{\mathrm{priv}}(A) := \mathbf{Adv}_F^{\$}(A) = \mathsf{Pr}_K[A^{F_K} = 1] - \mathsf{Pr}_{\$}[A^{\$} = 1].$$

We include initial value IV as a part of associated data D and so for nonce-respecting adversary A (never repeats a nonce or initial value and hence the view obtained by the adversary is nonce-respecting) the response of ideal oracle for every query is random as all queries are fresh. Similarly, we define online privacy for which the the ideal online authenticated encryption oracle $\$_{ol}$ responses random string keeping the online property. The online privacy advantage of an adversary A against F is defined as $\mathbf{Adv}_F^{\mathrm{opriv}}(A) := \mathbf{Adv}_F^{\$_{ol}}(A)$.

View and A-realizable. We define view of a deterministic adversary A interacting with an oracle $\mathcal{O}$ by a tuple $\tau(A^{\mathcal{O}}) := (Q_1, R_1, \ldots, Q_q, R_q)$ where Q_i is the i^{th} query and R_i is the response by $\mathcal{O}$. It is also called $\mathcal{O}$-view. A tuple $\tau = (Q_1, R_1, \ldots, Q_q, R_q)$ is called A-realizable if it makes query Q_i after obtaining all previous responses $R_1, \ldots, R_{i-1}$. As A is assumed to be deterministic, given $R_1, \ldots, R_q$, there is an unique q-tuple $Q_1, \ldots, Q_q$ for which the combined tuple is A-realizable. Now we describe the popular coefficient H-technique which can be used to bound distinguish advantage. Suppose f and g are two oracles and V denotes all possible A-realizable views while A interacts with f or g (they have same input and output space).

Lemma 1 (Coefficient H Technique). *If $\forall v \in V_{good} \subseteq V$ (as defined above), $\mathsf{Pr}[\tau(A^g(\cdot)) = v] \geq (1-\epsilon)\mathsf{Pr}[\tau(A^f(\cdot)) = v]$, then the distinguishing advantage $\mathbf{Adv}_g^f(A)$ of A is at most $\epsilon + \mathsf{Pr}[\tau(A^f(\cdot)) \notin V_{good}]$.*

We skip the proof as it can be found in many papers, e.g. [29].

2.2 Authenticity

We say that an adversary A **forges** an authenticated encryption F if A outputs (D, C, T) where $F_K(D, C, T) \neq \perp$ (i.e. it accepts and returns a plaintext), and A made no earlier query (D, M) for which the F-response is (C, T). It can make s attempts to forge after making q queries. We define that A forges if it makes at least one forges in all s attempts and the **authenticity-advantage** of A by

$$\mathbf{Adv}_F^{\mathrm{auth}}(A) = \mathsf{Pr}_K[A^{F_K} \text{ forges}].$$

Suppose for any valid tuple of associate data and tagged ciphertext (D, C, T), the tag T can be computed from (D, C). We write $T = T_K(D, C)$. So (D, C, T) is a valid tagged ciphertext if and only if $T_K(D, C) = T$. Almost all known authenticated encryptions F (including those following encrypt-then-mac paradigm) have this property for a suitably defined ciphertext C and tag function T. We know that PRF implies Mac. We use similar concept to bound authenticity. More formally, for any forgery B, there is a distinguisher A such that

$$\mathbf{Adv}_F^{\mathrm{auth}}(B) \leq \mathbf{Adv}_{(F,T)}^{\mathcal{O},\$}(A) + \frac{s}{2^n} \qquad (1)$$

where $\mathcal{O}$ and \$ are independent oracles and \$ is a random function. This can be easily seen by defining A as follows:

- A first makes the q many F-queries (D_i, M_i) which are made by B and obtains responses (C_i, T_i), $1 \leq i \leq q$.

- Then it makes s many T-queries (D_j, C_j), $q < j \leq q+s$ where (D_j, C_j, T_j)'s are returned by B.

- A returns 1 (interpreting that interacting with real) if and only if $T(D_j, C_j) = T'_j$ for some j.

The distinguishing advantage of A is clearly at least $\mathsf{Pr}[B \text{ forges}] - \frac{s}{2^n}$ and hence our claim follows.

Trivial Queries. As $F(D, M) = (C, T)$ implies that $T(D, C) = T$, we call such T-query (D, C) trivial (after obtaining response (C, T) response of the F-query (D, M)). The repetition of queries are also called trivial. Without loss of generality, we assume that all adversaries A is **deterministic and does not make any trivial query**. This assumptions are useful to simplify the analysis.

Block-wise Adversary. Due to limited memory in some environment such as low end devices the decryption oracle has to release a part of the plaintext before it authenticates. That raises some attacks on popular constructions [21]. We consider similar advantages such as privacy and authenticity, however the adversaries would have access of partial decryption oracles for authenticity security. Adding a layer of masking had been proposed as a countermeasure against these adversaries [14] at the cost of a streamcipher invocation. Here we introduce intermediate tag which also resist these adversaries at the cost of band-width.

3 ELmE: An Online Authenticated Encryption Algorithm

In this section, we demonstrate our new construction ELmE. It is an online authenticated encryption which takes an associated data $D \in \mathbb{B}^d$ and a messages $M \in \mathbb{B}^e$ and returns a tagged-ciphertext $C \in \mathbb{B}^{e+1}$ for all integers $d \geq 1$, $e \geq 1$. We assume associated data to be non-empty. The case when the associated data is empty, is taken care in the remark 2. To process incomplete blocks, one can either apply an injective padding rule (e.g., first pad 1 and then a sequence of zeros to make the padded message or associate data size multiple of n) or some standard methods (e.g., ciphertext stealing [10], the method used in Hash Counter Hash type constructions [11], XLS [31] etc.). It uses Encrypt-Mix-Encrypt type construction with a specified simple linear mixing (see in Algorithm 1) and a keyed block cipher $E_k : \mathbb{B} \to \mathbb{B}$ for the ECB layers. The ECB layers are masked by separate keys L_1 (for associated data), L_2 (for the message) and L_3 (for the ciphertext) chosen uniformly from $\mathbb{B}$. However, L_1, L_2, L_3 can be simply computed from E_k as $E_K(0) = L_1$, $E_K(1) = L_2$, $E_k(2) = L_3$ and can be preprocessed. Thus, for notational simplicity and simplifying security analysis, we demonstrate our constructions for complete block messages and with three independent keys L_1, L_2 and L_3. The complete construction is described below in Algorithm 1 and illustrated in Fig. 3 below. Very recent construction COPA (so does OCB) has also bottleneck in processing associate data. Clearly, ELmE has no such bottleneck.

Remark 1 (Similarity in Encryption and Decryption). The second ECB layer is based on blockcipher decryption instead of encryption. Due to this, both encryption and decryption behave almost in a similar fashion (only with few changes in masking layers due to different keys and in linear mixing which should be inverse of the forward mixing). This would help us to implement both encryption and decryption in hardware with a smaller area. Nowadays in all application environment, both encryption and decryption of blockciphers to be implemented and hence we can share the architectures to have a compact combined hardware implementation of it.

Remark 2 (Case when Associated data is zero). When the associated data is non zero, using the initial value of the sequence $W[0] = 0$, one can have a trivial attack against the privacy of the construction : Query any message M_1 with

```
**Input**: $(D, M) \in \mathbb{B}^d \times \mathbb{B}^e$
**Output**: $Z = (C, T) \in \mathbb{B}^e \times \mathbb{B}$
**Algorithm** ELmE$(D, M)$ (**Key**: $(L_1, L_2, L_3, K)$)
parse $D$ and $M$ into $n$-length blocks.
1     $D = D[1] \| \cdots \| D[d]$
2     $M = M[1] \| M[2] \| \cdots \| M[e]$
3     $W[0] = 0$
4     $M[e+1] = D[1] + \cdots + D[d] + M[1] + \cdots + M[e]$   (**checksum**)
process $D$
5     **For all** $j = 1$ **to** $d$
6         $DD[j] = D[j] + \alpha^{j-1}.L_1$ (**Masking the associate data blocks**)
7         $Z[j] = E_K(DD[j])$   (**Layer-I Encryption**)
8         $(Y'[j], W[j]) \leftarrow \rho(Z[j], W[j-1])$   (**Linear Mixing**)
process $M$
9     **For all** $j = 1$ **to** $e$
10        $MM[j] = M[j] + \alpha^{j-1}.L_2$ (**Masking the message blocks**)
11        $X[j] = E_K(MM[j])$   (**Layer-I Encryption**)
12        $(Y[j], W[d+j]) \leftarrow \rho(X[j], W[d+j-1])$   (**Linear Mixing**)
13        $CC[j] = E_K^{-1}(Y[j])$   (**Layer-II Encryption**)
14        $C[j] = CC[j] + \alpha^{j-1}.L_3$ (**Masking the ciphertext blocks**)
Tag generation
15    $MM[e+1] = M[e+1] + \alpha^e.L_2$
16    $X[e+1] = E_K(MM[e+1])$
17    $(Y[e+1], W[d+e+1]) \leftarrow \rho(X[d+e+1], W[d+e])$
18    $TT = E_K^{-1}(Y[e+1] + 0^{n-1}1)$
19    $T = TT + \alpha^e.L_3$
20    **Return** $(C = C[1] \| C[2] \| \cdots \| C[e], T)$

**Subroutine** $\rho(x, w)$ Onlinear Linear Mixing Function
21    $y = x + (\alpha + 1) \cdot w$
22    $w = x + \alpha \cdot w$
23    **Return** $(y, w)$
```

Algorithm 1: ELmE Authenticated Encryption Algorithm. Here α is a primitive element of the binary field $(GF(2^n), +, .)$.

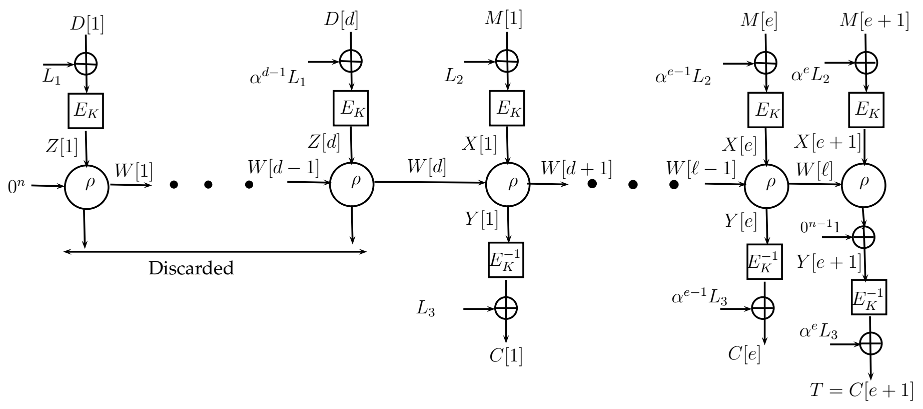

Fig. 3.1. Construction of ELmE Authenticated Encryption

$M_1[1] = 0$. It produces the ciphertext with $C_1[1] = L2 + L3$. Now querying any message M_2 with $M_2[1] = C_1[1]$ will produce $C_2[1] = 0$ with probability 1.

To resist against such attack, whenever associated data is empty, we initialize the value $W[0]$ to 1.

3.1 Underlying Layered Construction :

In this section we view the construction in a modular way which actually helps in understanding the design rational of our construction. Moreover, it also helps to understand the security analysis we will make later. Let mix be an online linear function, described below. We construct an online permutation based on the mix, a permutation $\pi : \mathbb{B} \to \mathbb{B}$ and masking functions $g_j : \mathbb{N} \times \mathbb{B} \to \mathbb{B}$, $j = 1, 2, 3$, such that $g_j(i, \cdot)$ is a permutation (we denote the inverse by $g_j^{-1}(i, \cdot)$). We take the usual masking functions $g_j(i, x) = \alpha^{i-1} \cdot L_j \oplus x$, $j = 1, 2, 3$. Let α is a primitive element of the field and $L_1 = E_k(0), L_2 = E_K(1)$ and $L_3 = E_K(2)$ (here we assume for simplicity that L_i's are uniform and independent to the underlying blockcipher).

- Layer-1: $DD[j] = g_1(j, D[j])$, $1 \le j \le d$, $MM[j] = g_2(j, M[j])$, $1 \le j \le e + 1$.
- Layer-2: $Z[i] = \pi(D[i])$, $1 \le i \le d$, $X[j] = \pi(MM[j])$, $1 \le j \le e + 1$.
- Layer-3: $Y = \mathsf{mix}(Z, X)$.
- Layer-4: $CC[j] = \pi^{-1}(Y[j])$, $1 \le j \le e$. $TT = \pi^{-1}(Y[e + 1] + 0^{n-1}1)$.
- Layer-5: $C[j] = g_3^{-1}(j, CC[j])$, $1 \le j \le e$. $T = g_3^{-1}(e + 1, TT)$.

mix Function : The mix function, we use is following :

$$Y = \begin{pmatrix} B_1 & B_2 \end{pmatrix} . \begin{pmatrix} Z[..d] \\ X[..e+1] \end{pmatrix}$$

where B_1 is a $((e+1)\times d)$ full matrix and B_2 is a $((e+1)\times(e+1))$ lower triangular invertible matrix. In particular, we choose a mix function as defined below for $1 \le i \le e+1$.
When $d \neq 0$:

$$\begin{aligned} Y[i] = {} & \alpha^{d+i-2}(\alpha+1)Z[1] + \cdots + \alpha^{i-1}(\alpha+1)Z[d] \\ & + \alpha^{i-2}(\alpha+1)X[1] + \alpha^{i-3}(\alpha+1)X[2] + \cdots + (\alpha+1)X[i-1] + X[i] \end{aligned}$$

When $d = 0$:

$$Y[i] = \alpha^{i-2}(\alpha+1)X[1] + \alpha^{i-3}(\alpha+1)X[2] + \cdots + (\alpha+1)X[i-1] + X[i] + \alpha^{i-1}(\alpha+1)$$

4 Privacy of ELmE

In this section we prove online privacy of ELmE. Thus it provides full privacy against all nonce-respecting adversaries. Let A be an adversary which makes q queries (D_i, M_i) and obtains responses (C_i, T_i), $1 \le i \le q$. We denote $\|D_i\| = d_i$, $\|M_i\| = \|C_i\| = e_i$ and $\|T_i\| = 1$. Let $\ell_i = d_i + e_i$ and $\sigma_{\text{priv}} = \sum_{i=1}^{q}(\ell_i + 1)$ (the total number of blocks in queries in addition with the checksum block). Let us fix an adversary A. Let $\$_{perm}$ denotes the random n-bit permutation and $\eta := \max_B \mathbf{Adv}_{E,E^{-1}}^{\$_{perm},\$_{perm}^{-1}}(B)$ denotes the maximum advantage over all adversaries B making at most σ_{priv} queries and running in time T_0 which is about time of the adversary A plus some overhead which can be determined from the hybrid technique.

Theorem 1.

$$\mathbf{Adv}^{\text{opriv}}_{\mathsf{ELmE}_{\Pi,\mathbf{L}}}(A) \ \le\ \frac{5\sigma_{\text{priv}}^2}{2^n}, \qquad \mathbf{Adv}^{\text{opriv}}_{\mathsf{ELmE}_{E_K,\mathbf{L}}}(A) \ \le\ \eta + \frac{5\sigma_{\text{priv}}^2}{2^n}.$$

The second part of the theorem is the standard hybrid argument. The first part follows directly from the coefficient H technique (see Lemma 1 and following Propositions 1 and 2. For this, we first need to define a set of good views V_{good} which would be applied in the proposition. Let us fix q message and associate data pairs $P_1 = (D_1, M_1), \ldots, P_q = (D_q, M_q)$ with $\|D_i\| = d_i, \|M_i\| = e_i$, $\ell_i = d_i + e_i$ and $\sigma = \sum_i \ell_i$. We denote $(P_1, \ldots, P_q)$ by τ_{in}. We assume that all P_i's are distinct.

Definition 1 (Good views). *A tagged ciphertext tuple $\tau_{out} = (C_1, \ldots, C_q)$ (also the complete view $\tau = (\tau_{in}, \tau_{out})$) is called* **good** *online view (belongs to τ_{good}) w.r.t. τ_{in} if (τ_{in}, τ_{out}) is an online view (i.e., it must be realized by an online cipher, see section 2) and the following conditions hold:*

1. *$C_i[j] = C_{i'}[j]$ implies that $D_i = D_{i'}$, $M_i[..j] = M_{i'}[..j]$ and*
2. *$\forall\ (i, l_i + 1) \neq (i', j')$, $T_i \neq C_{i'}[j']$.*

The first condition says that we can have collision of ciphertext blocks in a position only if they are ciphertexts of two messages with same prefixes up to that block. The second conditions says that all tag blocks are fresh as if these are independently generated. The following result says that in case of ideal online cipher, generating a bad view (i.e. not a god view) has negligible probability.

Proposition 1 (Obtaining a Good view has high probability).

$$\mathit{Pr}[\tau(A^{\$_{ol}}) \notin V_{good}] \;\leq\; \frac{\sigma^2_{\text{priv}}}{2^n}.$$

Proof. According to the definition, an online view is not a good view if $\exists i, j, i', j'$ with $C_i[j] = C_{i'}[j']$, where $(D_i, M_i[..j]) \neq (D_{i'}, M_{i'}[..j'])$. Suppose $i < i'$ or $i = i', j < j'$. Then $C_i[j]$ is computed by $(D_i, M_i[..j])$ before the computation of $C_{i'}[j']$. As $(D_i, M_i[..j]) \neq (D_{i'}, M_{i'}[..j'])$, the outcome of $C_{i'}[j']$ is random and fresh from $C_i[j]$. So, the probability that $C_i[j]$ takes the previously computed fixed value $C_i[j]$ is $\frac{1}{2^n}$. As at most $\binom{\sigma_{\text{priv}}}{2}$ pairs are there, the probability that $\tau(A^{\$^{OL}}) \notin V_{good}$ is at most $\frac{\sigma^2_{\text{priv}}}{2^n}$. □

We now fix a good view $\tau = (\tau_{in}, \tau_{out})$ as mentioned above. The tagged ciphertext of P_i is given by C_i which has $e_i + 1$ blocks where the last block $T_i := C_i[e_i + 1]$ denotes the tag. In the following result, we compute the interpolation probability, i.e. $\mathsf{Pr}[\tau(A^F) = \tau]$.

Proposition 2 (High interpolation probability of ELmE). $\forall \tau \in V_{good}$,

$$\mathit{Pr}[\tau(A^{\mathsf{ELmE}_{\Pi,\mathbf{L}}}) = \tau] \geq (1 - \tfrac{4\sigma^2_{\text{priv}}}{2^n}) \times \mathit{Pr}[\tau(A^{\$^{ol}}) = \tau].$$

Note that $\mathsf{Pr}[\tau(A^{\$^{ol}}) = \tau] = 2^{-nP}$ where P denotes the number of non-empty prefixes of (D_i, M_i), $1 \leq i \leq q$ as for every different prefixes, $\$^{ol}$ assigns an independent and uniform ciphertext blocks.

Remark 3. If we define L_1, L_2 and L_3 from E_K then we need to revise the proof of the Proposition 4 to obtain a modified ϵ' in Proposition 2. The revision is mainly by defining more internal bad events that some of the Π inputs is 0,1 or 2 (the inputs are used to generate L-values). As this adds notational complexity and does not increase the order of advantage (except the constant factor will increase) we skip it for clarity throughout the paper.

4.1 Proof of Proposition 2

As adversary is deterministic, we restrict to those good views which can be obtained by A. Hence the probability $\mathsf{Pr}[\tau(A^{\mathsf{ELmE}}) = \tau]$ is same as

$$\mathsf{Pr}[\mathsf{ELmE}_{\Pi,\mathbf{L}}(D_i, M_i) = Z_i, 1 \leq i \leq q].$$

Before computing interpolation probability we denote all intermediate variables while computing $\mathsf{ELmE}_{L_1,L_2,L_3,\pi}(D_i, M_i) = C_i$. Let for all i and j whenever defined

1. $DD_i[j] = L_1 \cdot \alpha^{j-1} + D_i[j]$, $MM_i[j] = L_2 \cdot \alpha^{j-1} + M_i[j]$
2. $\Pi(DD_i[j]) = Z_i[j]$, $\Pi(MM_i[j]) = X_i[j]$,
3. $\mathsf{mix}(Z_i, X_i) = Y_i$
4. $CC_i[j] = L_3 \cdot \alpha^{j-1} + C_i[j]$ and finally $TT_i = L_3 \cdot \alpha^e + T_i$.

Note that CC and TT have been defined through tagged-ciphertext and L_3 instead of applying Π on Y blocks. Let $\mathbf{DD} = (DD_1, \ldots, DD_q)$ and similarly we define $\mathbf{MM}$, $\mathbf{Z}, \mathbf{X}, \mathbf{Y}$ and $\mathbf{CC}$. So, we have $\mathsf{mix}(\mathbf{Z}, \mathbf{X}) = \mathbf{Y}$ with the extended definition of mix which applies mix function for each (Z_i, X_i).

Collision Relation. Now we define a collision relation of a vector $(x_1, \ldots, x_t)$ by the equivalence relation $\mathsf{coll}(x)$ for which i is related to j if and only if $x_i = x_j$.

We call (L_1, L_2, L_3) valid if it computes $(\mathbf{DD}, \mathbf{MM}, \mathbf{CC}, \mathbf{TT})$ for which only equality among the blocks occurs in $SS_i[j] = SS_{i'}[j]$ where $S_i[j] = S_{i'}[j]$ (in case of $S = T$, we only have $j = 1$.

Lemma 2. *$Pr[(L_1, L_2, L_3)$ is valid$] \geq (1 - \epsilon_1)$ where $\epsilon_1 = \frac{2\sigma_{\text{priv}}^2}{2^n}$.*

Proof. We prove it by using union bound applied to all equality (which violates that (L_1, L_2, L_3) is valid). Because of primitiveness of α and uniform independent choice of L_1, L_2 and L_3 each equality violating valid has probability 2^{-n}. As there are at most $\binom{2\sigma_{\text{priv}}}{2}$ equality the result follows. □

Consistent collision relations for a linear function Suppose $X = X[1..r_1]$ be a r_1-tuple of variables of $\mathbb{B}$ and $L : \mathbb{B}^{r_1} \to \mathbb{B}^{r_2}$ be a linear function. We denote $Y = L(X)$ which is an r_2-tuple of variables from $\mathbb{B}$. Let γ_1 and γ_2 are two equivalence relations defined on the sets respectively $[1..r_1]$ and $[1..r_2]$. Let X^{γ_1} denote the tuple of variables which satisfies the collision relation γ_1 by replacing identical variables by the variable which occurred with minimum index. We say that (γ_1, γ_2) is consistent with L if $L_i(X^{\gamma}) \equiv L_j(X^{\gamma_1})$ if and only if i and j are related in γ_2. Clearly, given any γ_1 and L there is exactly one γ_2 for which (γ_1, γ_2) is consistent with L. We write $\gamma_1 \Rightarrow_L \gamma_2$.

Example 1. If $\gamma_1 = \{\{1,3\}, \{2\}, \{4,6\}, \{5\}\}$ for $r_1 = 6$, then we write $X^{\gamma_1} = (X_1, X_2, X_1, X_4, X_5, X_4)$. Let L map into three variables (i.e., $r_2 = 3$ such that $L_1 = X_1 + X_2 + X_3 + X_6$, $L_2 = X_4 + X_5 + X_6$ and $L_3 = X_2 + X_4$ then $L_1(X^{\gamma_1}) = L_3(X^{\gamma_1}) = X_2 + X_4$ and $L_2(X^{\gamma_1}) = X_5$ (we work it here in binary field). So $\gamma_1 \Rightarrow_L \gamma_2$ where $\gamma_2 = \{\{1,3\}, \{2\}\}$.

Lemma 3. *[Number of Solutions for Consistent relations] Let (γ_1, γ_2) be consistent with $L : \mathbb{B}^{r_1} \to \mathbb{B}^{r_2}$ then*

$$|\{X : Coll(X) = \gamma_1, Coll(L(X)) = \gamma_2\}| \geq 2^{ns_1} \times (1 - \frac{s^2}{2^{n+1}})$$

where s_1 and s_2 denote the number of equivalence classes of γ_1 and γ_2 respectively and $s = s_1 + s_2$.

Proof. Let $Y = L(X)$. Because of consistency, for all related i, j in γ_2, $Y_i = Y_j$. There may be additional equality which must be avoided. For all unrelated pair (i, j) in γ_2 we must choose X in a manner such that $Y_i \neq Y_j$ and similarly for all unrelated pair (i, j) in γ_1 we have $X_i \neq X_j$. Due to consistency, any one can happen for at most $2^{n(s_1-1)}$ many X's as $L_i(X^{\gamma_1}) = L_j(X^{\gamma_1})$ gives a non-trivial equation. So the result follows as we have at most $\binom{s}{2}$ such equalities. □

Now we establish two collision relations γ_1 and γ_2 which are consistent with the linear mix function. These relations are defined based on a good view τ. Let γ_1 be the collision relation defined on the set $\{(i, j, M) : i \leq q,\ j \leq l_i + 1\} \bigcup \{(i, j, D) : i \leq q, j \leq d_i\}$. A pair $((i, j, S), (i', j', S'))$ is related if $S = S'$, $j = j'$ and $S_i[j] = S_{i'}[j]$. All other pairs are unrelated. Let γ_2 be the a collision relation defined on the set $\{(i, j, C) : i \leq q,\ j \leq l_i + 1\}$ for which only pairs $((i, j, C), (i', j', C))$ if $j = j'$ and $C_i[j] = C_{i'}[j]$. Let the no. of equivalence class of γ_i be s_i, $i = 1, 2$. Note that $s_2 = P$, the number of prefixes of (D_i, M_i) containing at least one message block.

Lemma 4. *The collision relations defined as above is consistent with* mix.

Proof. Let $\mathbf{Y} = (Y_1 := \mathsf{mix}(Z_1, X_1), \ldots, Y_q := \mathsf{mix}(Z_q.X_q))$. Since the view is good, $C_i[j] = C_{i'}[j]$ can happen if $D_i = D_{i'}$ and $M_i[..j] = M_{i'}[..j]$. In this case, clearly, $Y_i[j] = Y_{i'}[j]$. Now for any other pair $((i, j), (i', j'))$, it is easy to see that mix function leads to a non-trivial equation $\mathsf{mix}_j(X_i^{\gamma_1}) = \mathsf{mix}_{j'}(X_{i'}^{\gamma_1})$. □

Corollary 1. $\#\{(Z, X) : \mathit{coll}(Z, X) = \gamma_1, \mathit{coll}(Y) = \gamma_2\} \geq 2^{ns_1}(1 - \frac{2\sigma_{\text{priv}}^2}{2^n})$

Now, for a fixed valid-L triple (L_1, L_2, L_3), the conditional interpolation probability is

$$\sum_{(Z,X)} \frac{\#\pi : \pi(MM) = X, \pi(DD) = Z, \pi(CC) = Y}{\#\pi} \geq (1 - \frac{2\sigma_{\text{priv}}^2}{2^n}) \times 2^{-nP}.$$

So by multiplying the probability for validness of (L_1, L_2, L_3) the proof of the proposition completes.

5 Authenticity of ELmE

Let $\mathbf{L} = (L_1, L_2, L_3)$ be the triple of masking keys and Π be the uniform random permutation. For notational simplicity, we write $\mathsf{ELmE}_{\Pi,\mathbf{L}}$ by F. Note that for a valid tuple of associate data and tagged ciphertext (D, C, T), the tag T can be computed from C and the key. We write $T = T_{\Pi,\mathbf{L}}(D, C) := T(D, C)$. So (D, C, T) is a valid tagged ciphertext if and only if $T(D, C) = T$. As we have observed in Eq. 1, we only need to show indistinguishability for which we apply the coefficient H technique again. For this, we need to identify set of good views for which we have high interpolation probability.

Good View. A (F, T)-view of a distinguisher A is the pair $v = (\tau_F, \tau_T)$ where $\tau_F = (D_i, M_i, C_i, T_i)_{1 \le i \le q}$ is an q-tuple of F-online view and $\tau_F = (D_j, C_j, T_j)_{q < j \le q+s}$ is an s-tuple non-trivial T-view. It is called **good** if τ_F is good (as defined in Definition 1) and for all $q < j \le q + s$, T_j's are fresh - distinct and different from all other T_i's and $C_i[j]$'s. We recall the notation $|M_i| = e_i$, $|D_i| = d_i$ and $\ell_i = d_i + e_i$. Let $\sigma_{\text{auth}} = \sum_{i=1}^{q+s}(\ell_i + 1)$. Since F is online function we consider pair of independent oracles $(\$_{ol}, \$)$ where $\$_{ol}$ denotes the random online function and \$ is simply a random function.

Proposition 3 (Realizing good view while interacting with random function has high probability). *For all adversary A,*

$$\mathsf{Pr}[\tau(A^{\$_{ol},\$}) \text{ is not good }] \le \frac{(q + \sum_{i=1}^{q} e_i)^2}{2^{n+1}} + \frac{s(q + s + \sum_{i=1}^{q+s} e_i)}{2^n} \le \frac{2\sigma_{\text{auth}}^2}{2^n}.$$

As in Proposition 1, we can similarly prove the above. The first summand takes care the collisions in $C_i[j]$'s (i.e., the bad view for τ_F as in Proposition 1) and the second summand takes care the collision between T_i's $(q < i \le q+s)$ and all other $C_i[j]$'s. Now we fix a good view $\tau = (\tau_F, \tau_T)$ as defined above (following same notations). Now it is easy to see that obtaining τ interacting with $(\$_{ol}, \$)$ has probability $2^{-ns} \times 2^{-n\sigma_{pf}} = 2^{-n(s+\sigma_{pf})}$ where σ_{pf} denotes the number of non-empty prefixes of (C_i, T_i), $1 \le i \le q$ (at those blocks random online function returns randomly).

Proposition 4 (Good view has high interpolation probability). *For any good (F, T)-view τ and $\epsilon' = 5\sigma_{\text{auth}}^2/2^n$, we have*

$$\mathsf{Pr}[F(D_i, M_i) = (C_i, T_i), 1 \le i \le q,\ T(D_j, C_j) = T_j, q < j \le q+s] \ \ge\ (1-\epsilon')2^{-n(\sigma_{pf}+s)}.$$

The proof is given in the following section. Assuming this, the pair (F, T) is ϵ-indistinguishable from $(\$_{ol}, \$)$ with $\epsilon = \epsilon' + 2\sigma_{\text{auth}}^2/2^n$. So we have proved the following theorem.

Theorem 2. *Let an adversary A makes q distinct queries and attempts to forge against the $\mathsf{ELmE}_{\Pi,\mathbf{L}}$ at most s times. Then the forging advantage*

$$\mathbf{Adv}^{\text{forge}}_{\mathsf{ELmE}}(\mathcal{A}) \ \le\ \frac{7\sigma_{\text{auth}}^2}{2^n} + \frac{s}{2^n}.$$

So the authenticity of the blockcipher based construction $\mathsf{ELmE}_{K,\mathbf{L}}$ is at most $\mathbf{Adv}^{\$_{perm},\$_{perm}^{-1}}_{E,E^{-1}}(\sigma_{\text{auth}}, T_0) + \frac{7\sigma_{\text{auth}}^2}{2^n} + \frac{s}{2^n}$ where T_0 denotes the time of A plus some overhead.

5.1 Proof of Proposition 4

We choose $X_1, \ldots, X_q$ and then $Y_{q+1}, \ldots, Y_{s+q}$ which fix all internal X and Y values except the last block for the s many T-queries. We explicitly provide counting steps by steps. We choose valid L which fixes MM's for the first q messages and, CC's and DD's for all $s + q$ queries. We can then choose MM for these s queries so that checksums are all fresh and for all these fresh checksums we can ensure last Y blocks fresh by choosing X blocks appropriately. Now we make these choices one by one more formally.

Choices of Valid L-triples We first define valid L-triples as defined in privacy. A triple (L_1, L_2, L_3) is called valid w.r.t. the fixed good (F, T)-view τ if the computed MM, DD, CC and TT values satisfy the collision relations described below and whenever C_j, $j > q$, is a strictly prefix of C_i, $i \leq q$ and $D_i = D_j$ then $MM_i[e_i] \neq MM_j[e_j]$, i.e., equivalently $M_i[e_j + 1] + \ldots M_i[e_i] + L_2(\alpha^{e_j} + \ldots \alpha^{e_i}) \neq 0$. To define the collision (equivalence) relation, we mention those places where equivalence occurs. In all other places these are not related. $SS_i[j] \equiv SS_{i'}[j]$ if $S_i[j] = S_{i'}[j]$ where S represents any one of the four symbols M, D, C and T. So they can be identical only if their positions as well as symbols (or types of the input) match. The simple counting argument with union bound applied to all individual bad events proves the following result.

Lemma 5. $\mathsf{Pr}[(L_1, L_2, L_3)$ *is a* ***L-valid*** *triple*$] \geq (1 - \frac{2\sigma_{\text{auth}}^2}{2^n})$.

Choices of valid $\boldsymbol{Z}, \boldsymbol{X}, \boldsymbol{Y}$ except the last blocks for the last s queries As in section 4, τ_F induces consistent collision relations of $(\mathbf{Z}, \mathbf{X}) := (\mathbf{Z}_1, \ldots, \mathbf{Z}_q, \mathbf{X}_1, \ldots, \mathbf{X}_q)$ and $\mathbf{Y} := (\mathbf{Y}_1, \ldots, \mathbf{Y}_q)$. Now we extend this collision relation to $(\mathbf{Z}_{q+1}, \mathbf{Y}_{q+1}, \ldots, \mathbf{Z}_{q+s}, \mathbf{Y}_{q+s})$ as follows for $j < i \leq q + s$:

1. $\mathbf{Z}_i[j] \equiv \mathbf{Z}_{i'}[j']$ if $j = j'$ and $D_i[j] \equiv D_{i'}[j]$.
2. $\mathbf{Y}_i[j] \equiv \mathbf{Y}_{i'}[j']$ if $j = j'$ and $C_i[j] \equiv C_{i'}[j]$.

The collision relation on $(\mathbf{Z}, \mathbf{Y})$ induces a collision relation on $\mathbf{X}_f := (\mathbf{X}_{q+1}, \ldots, \mathbf{X}_{q+s})$ through the linear mix^{-1} function. That is, $(\mathbf{Z}, \mathbf{Y}) \Rightarrow_{\mathsf{mix}^{-1}} \mathbf{X}_f$. Let γ_1' be the extended collision relation on $(\mathbf{Z}, \mathbf{X})$ and γ_2' be that of Y. We denote the number of equivalence classes by s_1' and s_2'. By using the counting on consistency relations (see Lemma 3) the number of (Z, X, Y) with $\mathsf{mix}(Z, X) = Y$ and $\mathsf{coll}(Z, X) = \gamma_1'$, $\mathsf{coll}(Y) = \gamma_2'$ is at least

$$2^{n(s_1+s_3)}(1 - \frac{(s_1' + s_2')^2}{2^{n+1}}) \geq 2^{n(s_1+s_3)}(1 - \frac{\sigma_{\text{auth}}^2}{2^{n+1}})$$

where s_3 denotes the number of additional equivalence classes in $\mathbf{Y}_f$ which are not present in $(\mathbf{Y}_1, \ldots, \mathbf{Y}_q)$. Thus, s is the number of blocks we can choose freely which determines all other blocks. Now we state an important property of these collision relations γ_1' and γ_2'.

Lemma 6. *If for some $j > q$, $\forall k \leq \ell_j$, $X_j[k] \equiv X_{r_k}[k]$, $r_k \leq q$ then $\forall k \leq \ell_j$, $X_j[k] \equiv X_i[k]$ for some $i \leq q$. This means the message corresponding to a forged ciphertext is the prefix of some other messages, queried previously by the adversary.*

Proof. Let us fix $j = q + 1$ (for all other j, the argument is similar) and denote ℓ_j by ℓ. Now we have the following identities: $X_{q+1}[k] \equiv X_{r_k}[k]$ for all k. This can happen only if $Y_{q+1}[j] \equiv Y_{t_j}[j]$ for some $t_j \leq q$, otherwise $X_{q+1}[j]$ would get completely new variable which is not present in all first q queries. Now if we write $X_{q+1}[j]$ in terms of these X_{t_j}'s variable one can obtain the desired result. $\square$

Choices of MM for forging s queries Given the choices of valid L and those of X, Y, Z as described above we can now choose remaining MM values satisfying same collision relation as $(X_{q+1}, \ldots, X_{q+s})$. More precisely, we can choose all those MM values for which $X_j[i]$'s are fresh. Let s_4 denote the number of additional distinct blocks in $(X_{q+1}, \ldots, X_{q+s})$ which are not present in $(X_1, \ldots, X_q)$. The number of these s_4 blocks MM different from all other defined MM, DD and CC blocks such the all last blocks of MM_j's $(j > q)$ are fresh is at least $2^{ns_4}(1 - \frac{2\sigma^2_{\text{auth}}}{2^n})$. Note that $MM_i[e_i + 1] = MM_{i'}[e_{i'} + 1]$ induces a restriction on choices of MM.

Choices of last block of X for these s queries For any such previous choices, we now choose the blocks of $X_j[e_j + 1]$, $j > q$ so that the last block of Y_j's are fresh. This can be chosen in $2^{ns}(1 - \sigma^2_{\text{auth}}/2^n)$ ways.

Armed with all these counting, the interpolation probability is at least

$$(1 - 5\sigma^2_{\text{auth}}/2^n) \times 2^{-n(\sigma_{pf}+s)}.$$

This completes the proof.

6 Our Construction incorporating Intermediate Tags

Suppose, we want ELmE with intermediate tags generated after each it blocks. In this case, for a message $M \in \mathbb{B}^e$, ELmE generates a ciphertext $C \in \mathbb{B}^e$ and $T \in \mathbb{B}^h$ where $h = \lceil \frac{e}{k} \rceil$. Processing of D remains same. For Processing of M, the calculation of $C[j]$ is changed to $CC[j] + \alpha^{j-1+\lceil \frac{j-1}{k} \rceil}.L_3$. $\forall$ $j < e$ s.t. $k|j$, the intermediate tags are generated by $T[\frac{j}{k}] = E_K^{-1}(W[d+j]) + \alpha^{j-1+\lceil \frac{j-1}{k} \rceil}.L_3$. Final tag $T[h]$ is generated similar to the generation of T in the case of ELmE without intermediate tags (Here $\alpha^{e+h-1}L_3$ is used as the mask). Tag T is given by $T[1] \mid\mid T[2] \mid\mid \cdots \mid\mid T[h]$. For verification during decryption, each $T[i]$ is verified and as soon as, a $T[i]$ doesn't matched with it's calculated value, the ciphertext gets rejected.

Intermediate tags can be used in authenticated encryption to provide quick rejection of invalid decryption queries. This also helps in low-end implementation where the message has to be released depending on buffer size. If we have an intermediate tag in appropriate positions so that we can reject before we release some message blocks. Our construction can be easily extended to produce intermediate tags also, as described in the figure below. Here, we have used intermediate tags after processing of each $k \leq n$ blocks of message.

6.1 Attack against the construction when $k > n$

Here is an demo example of how the construction works when $n = 4$. We have shown it for $k > 4$ with a degree 4 primitive polynomial $x^4 + x^3 + 1$. For simplicity

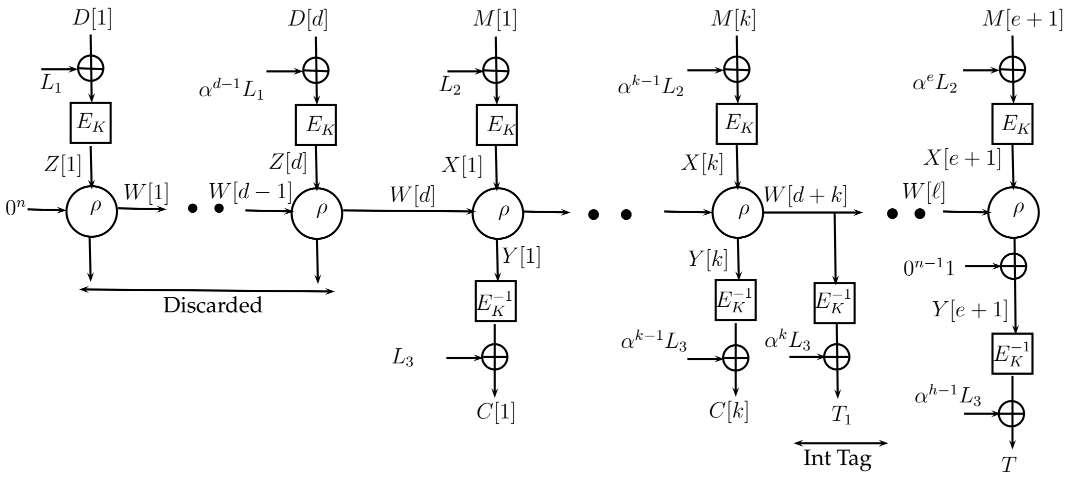

Fig. 6.1. ELmE with intermediate tags

we take empty associated data. The attack is described below. The associated data part is considered as null for all the queries however the attack works for any fixed associated data.

1. **query-1**: $F_K(M[1..6]) = (C[1..5], T[1], C[6], T[2])$.
2. **query-2**: $F_K(M'[1], M[2..6]) = C'[..5], T'[1], C'[6], T'[2])$.
3. **forged ciphertext**: $(C[1], C'[2..4], C[5], T'[1] \cdots)$.

This follows from the following equality for the input of the blockcipher invocation which computes the intermediate tag:

$$\begin{aligned} W_F[5] &= X_2[5] + \alpha X_2[4] + \alpha^2 X_2[3] + \alpha^3 X_2[2] + ((\alpha^4+\alpha^3+1)X_1[1] + (\alpha^3+1)X_2[1]) \\ &= X_2[5] + \alpha X_2[4] + \alpha^2 X_2[3] + \alpha^3 X_2[2] + \alpha^4 X_2[1] \text{ (As } \alpha^4+\alpha^3+1=0) \\ &= W_2[5] \end{aligned}$$

Main Idea: Suppose the forged ciphertext is $C_{i_1}[1] \cdots C_{i_5}[5]T_i[1]\cdots$ where $i_1, \cdots, i_5, i$ are the messages queried. Then we have, $W_F[5] = ((\alpha^4+\alpha^3)X_{i_5}[1] + (\alpha^3+\alpha^2)X_{i_4}[1] + (\alpha^2+\alpha)X_{i_3}[1] + (\alpha+1)X_{i_2}[1] + X_{i_1}[1]) + ((\alpha^3+\alpha^2)X_{i_5}[2] + \cdots + X_{i_2}[2]) + \cdots + X_{i_5}[5]$. To make this equation trivial with $W_i[5] = X_i[5] + \alpha X_i[4] + \alpha^2 X_i[3] + \alpha^3 X_i[2] + \alpha^4 X_i[1]$, we make j^{th} blocks of all the messages $i_1, \cdots, i_5, i$ to be same for all $j \geq 2$. Using the fact that α is a root of the primitive polynomial of degree 4, we will assign $X_j[1], X_{i_1}[1], \cdots, X_{i_5}[1]$ one of two values $X_1[1], X_2[1]$ such that (assigning all the values to one particular is not allowed) regardless of the exact values of the two, the following equation become trivial : $\alpha^4 X_j[1] = (\alpha^4+\alpha^3)X_{i_5}[1] + (\alpha^3+\alpha^2)X_{i_4}[1] + \cdots + (\alpha+1)X_{i_2}[1] + X_{i_1}[1]$. Assigning $X_{i_5}[1] = X_{i_1}[1]$ and $X_{i_4}[1] = X_{i_3}[1] = X_{i_2}[1] = X_i[1]$ we obtain that. It is easy to see that the equality of the X values ensures that, $i_5 = i_1$ and $i_4 = i_3 = i_2 = i$. Hence, we have just two queries. This idea can be similarly extended to have an attack against the construction when $it > n$ using the primitive polynomial of degree n.

6.2 Security of the construction when $k \leq n$

Let the forged Ciphertext be $C_{i_1}[1]C_{i_2}[2]\cdots C_{i_k}[k]T_i$. If $W_i[k] \equiv W_f[k]$ then we have the following set of equalities: $\forall j \leq k$,

$$(\alpha^{k-j}+\alpha^{k-j-1})X_{i_k}[j]+(\alpha^{k-j-1}+\alpha^{k-j-2})X_{i_{k-1}}[j]+\cdots(\alpha+1)X_{i_{j+1}}[j]+X_{i_j}[j] \equiv \alpha^{k-j}X_i[j]$$

This equation is trivial only if $\forall j \leq k,\ X_{i_k}[j] \equiv \cdots \equiv X_{i_j}[j] \equiv X_i[j]$, otherwise we have an polynomial of α of degree less than or equal to $k-j$ whose value is 0, which contradicts that α is a primitive element of $GF(2^{128})$. This makes $\forall j \leq k,\ X_{i_k}[j] \equiv X_i[j]$ meaning that the forged ciphertext block $C_f[1..k] = C_i[1..k]$, i^{th} ciphertext response. If the forged ciphertext and the i^{th} message's ciphertext is identical up to $k.k'$ blocks but not identical up to $k.(k'+1)$ blocks then also, using the primitiveness property we have $W[k.(k'+1)] = W[k.(k'+1)]$ is a non trivial equation. This observation can be extended to prove the privacy and authenticity of our construction with intermediate tag. We state the theorems for privacy and authenticity. Let F denote our construction with intermediate tags for every k blocks.

Theorem 3.

$$\mathbf{Adv}^{\text{opriv}}_{F_{\Pi,\mathbf{L}}}(A) \leq \frac{5\sigma^2_{\text{priv}}}{2^n}, \quad \mathbf{Adv}^{\text{opriv}}_{\mathsf{ELmE}_{E_K,\mathbf{L}}}(A) \leq \eta + \frac{5\sigma^2_{\text{priv}}}{2^n}.$$

Theorem 4. *Let an adversary A makes q distinct queries and attempts to forge against the $\mathsf{ELmE}_{\Pi,\mathbf{L}}$ at most s times. Then the forging advantage*

$$\mathbf{Adv}^{\text{forge}}_{\mathsf{ELmE}}(\mathcal{A}) \leq \frac{7\sigma^2_{\text{auth}}}{2^n} + \frac{s}{2^n}.$$

So the authenticity of the blockcipher based construction $\mathsf{ELmE}_{K,\mathbf{L}}$ is at most $\mathbf{Adv}^{\$_{perm},\$^{-1}_{perm}}_{E,E^{-1}}(\sigma_{\text{auth}}, T_0) + \frac{7\sigma^2_{\text{auth}}}{2^n} + \frac{s}{2^n}$ where T_0 denotes the time of A plus some overhead.

Due to page limitation we skip the proof and the proofs can be found in the full version.

6.3 Including Intermediate tags : Comparison with COPA

Intermediate tags are used to provide block-wise security. Suppose we consider a construction with intermediate tag size of k blocks. At each k blocks, we check the intermediate tag, hold the k block message and finally release the k blocks of the message if the tag is verified. For that, we need to store all the intermediate computations and the already computed messages in order to perform the verification. As we are using low end device, we need to minimize the buffer size.

Now, generating intermediate tags for COPA is not as straight forward as ELmE as similar approach won't provide any security because identical last two blocks will produce same intermediate tag.

Moreover, we claim that even if intermediate tags is produced for COPA as if the final tag, then it also has the disadvantage of requiring additional buffer storage. Now we compare the 20 round pipeline implementations which is keeping computing the messages even after intermediate tag to keep the pipeline full. For each k block of intermediate tags, the pipelined implementation of 20 round AES for COPA requires to store k block messages and in addition 20 blocks of intermediate values for the subsequent ciphertext blocks. On the other hand ELmE requires k blocks messages and 10 blocks of intermediate computation for next 10 next subsequent ciphertext. We save 10 blocks in buffer mainly due to faster verification (ELmE verifies after one layer, whereas COPA verifies after two layers). It has great advantage for low-end devices (keeping in mind that, block-wise adversaries are considered only when buffer size is limited implying low-end device).

Keeping the above benefits into consideration, we opt for the linear mix ρ function rather than using a simple xor operation, as used in COPA.

7 Conclusion and Future Works

7.1 Discussion on Performance

We mainly provide comparisons of OCB3, McOE-D, COPA and our construction ELmE. All the constructions have same key size and similar number of random mask (which can be preprocessed) for masking layers. The number of blockcipher calls for processing every message, associate data and tag blocks are given in the Table 1. The speed up for OCB, COPA and ELmE is p with parallel implementations by p processors as their construction support parallel execution. Due to the sequential nature of the lower level of McOE-D, the speed up factor can be at most 2.

Now, we briefly discuss bottlenecks issues of the other constructions. COPA has bottleneck in associated data and hence it requires additional waiting for obtaining intermediate values from associated data. $McOE$ uses $TC3$ type encryption and it's lower level has a CBC type structure which can not be executed in parallel implying the construction can not be pipelined. Hence it has a hardware bottleneck. $OCB3$ (which has minimum bottleneck among all versions) has a bottleneck in the IV processing. As the encryption of the IV is needed in the masking of the messages, hence the encryption of the messages can start only after the encryption of IV, hence has the bottleneck of having additional clock cycles required for one block encryption. Our construction is completely parallel with no such bottleneck as described above. Moreover the construction treats the additional data and message exactly in a similar way (except with different

Construction	#BC call per AD block	#BC call per Message block	#BC call per tag block	speed up	Misuse Resistance	Bottleneck
OCB	1	1	1	p	No	IV Processing
McOE	1	2	2	2	Yes	Lower level Processing
CoPA	1	2	2	p	Yes	Associated data Processing
ELmE	1	2	1	p	Yes	None

Table 1. Comparative study on the performance of block-cipher based Authenticated Encryptions

masking keys). The encryption and decryption also behave similarly and hence ensures less chip area in hardware implementation. Moreover our construction can incorporate intermediate tags (with intermediate tag length less than or equal to 128), which provides quick rejection of invalid decryption queries ensuring the construction's security even against block-wise adaptive adversaries.

7.2 Future Works

We have planned to submit our proposed authenticated cipher ELmE as a proposal in CAESER competition. So, we'll implement a portable reference software implementation of our cipher as well as include a reference hardware design in Verilog. Due to the parallel nature, we expect to get about p times speed-up by using p parallel processors. In hardware, a full round pipeline implementation of our construction based on AES blockcipher is expected to provide 128 bits of ciphertext/tag in every clock-cycle after the initial phase is over to get pipeline saturated.

8 Acknowledgement

This work is supported by the Centre of Excellence in Cryptology (CoEC), Indian Statistical Institute, Kolkata.

References

[1] — (no editor), *CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness*. URL: http://competitions.cr.yp.to/caesar.html.

[2] — (no editor), *DIAC : Directions in Authenticated Ciphers* (2012). URL: http://hyperelliptic.org/DIAC/.

[3] — (no editor), *DIAC : Directions in Authenticated Ciphers* (2013). URL: http://2013.diac.cr.yp.to/.

[4] — (no editor), *Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 and 128-EIA3. Document 2: ZUC Specification. ETSI/SAGE Specification, Version: 1.5* (2011). Citations in this document: §1.1.

[5] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar W. Tischhauser and Kan Yasuda, *Parallelizable (authenticated) online ciphers* (2013), *Asiacrypt (to be published)*, 2013. Citations in this document: §1.1.

[6] M.Bellare, P.Rogaway and D.Wagner, *The EAX Mode of Operation (A Two-Pass Authenticated Encryption Scheme Optimized for Simplicity and Efficiency)* **3017** (2004), 389–407, *Fast Software Encryption*, Lecture Notes in Computer Science, 2004. Citations in this document: §1.1.

[7] M.Bellare, J.Blake and P.Rogaway, *OCB : A Block-Cipher Mode of Operation for Efficient Authenticated Encryption* **6** (2005), 365–403.

[8] G.Bertoni, J.Daeman, M.Peeters and G.V.Assche, *Duplexing the Sponge : Single Pass Authenticated Encryption and Other Applications* **7118** (2011), 320–337, *Selected Areas in Cryptography*, Lecture Notes in Computer Science, 2011. Citations in this document: §1.1.

[9] Pompiliu Donescu and Virgil D. Gligor, *Fast Encryption and Authentication : XCBC Encryption and XECB Authentication Modes* **2355** (2001), 92–108, *Fast Software Encryption*, Lecture Notes in Computer Science, 2001. Citations in this document: §1.1.

[10] M. Dworkin, *Recommendation for block cipher modes of operation: three variants of ciphertext stealing for CBC mode. Addendum to NIST Special Publication 80038A.* (2010). Citations in this document: §3.

[11] Dengguo Feng, Peng Wang and Wenling Wu, *HCTR : A Variable-Input-Length Enciphering Mode* **3822** (2005), 175–188, *CISC*, Lecture Notes in Computer Science, 2005. Citations in this document: §3.

[12] N. Ferguson, D. Whiting, B. Schneier, J. Kelsey, S. Lucks and T. Kohno, *Helix, Fast Encryption and Authentication in a Single Cryptographic Primitive* (2003), *Fast Software Encryption*, 2003. Citations in this document: §1.1.

[13] E. Fleischmann, C. Forler, S. Lucks, *McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes* **7549** (2012), 196–215, *Fast Software Encryption*, Lecture Notes in Computer Science, 2012. Citations in this document: §1.1, §1.1.

[14] Pierre-Alain Fouque, Antoine Joux, Gwenaelle Martinet, and Frederic Valette., *Authenticated On-Line Encryption* **3006** (2003), 145–159, *Selected Areas in Cryptology*, Lecture Notes in Computer Science, 2003. Citations in this document: §2.2.

[15] Shai Halevi and Phillip Rogaway, *A Tweakable Enciphering Mode* **2729** (2003), 482–499, *CRYPTO*, Lecture Notes in Computer Science, 2003. Citations in this document: §1.1.

[16] Martin Hell, Thomas Johansson, Alexander Maximov and Willi Meier, *A Stream Cipher Proposal: Grain-128* (2005), *eSTREAM, ECRYPT Stream Cipher Project, Report 2006/071*, 2005. URL: `http://www.ecrypt.eu.org/stream`. Citations in this document: §1.1.

[17] Russ Housley, Doug Whiting and Niels Ferguson, *Counter with CBC-MAC (CCM).* (2003), *RFC 3610 (Informational)*, 2003. Citations in this document: §1.1.

[18] T. Iwata, *New Blockcipher Modes of Operation with Beyond the Birthday Bound Security* **4047** (2006), 310–327, *Fast Software Encryption*, Lecture Notes in Computer Science, 2006. Citations in this document: §1.1.

[19] Tetsu Iwata and Kan Yasuda, *HBS : A Single-Key mode of Operation for Deterministic Authenticated Encryption* **5665** (2009), 394–415, *Fast Software Encryption*, Lecture Notes in Computer Science, 2009. Citations in this document: §1.1.

[20] Tetsu Iwata and Kan Yasuda, *BTM : A Single-Key, Inverse-Cipher-Free Mode for Deterministic Authenticated Encryption* **5867** (2009), 313–330, *Selected Areas in Cryptography*, Lecture Notes in Computer Science, 2009. Citations in this document: §1.1.

[21] Antoine Joux, Gwenlle Martinet and Fredric Valette, *Blockwise-Adaptive Attackers: Revisiting the (In)Security of Some Provably Secure Encryption Models: CBC, GEM, IACBC* **2442** (2002), 17–30, *CRYPTO*, Lecture Notes in Computer Science, 2002. Citations in this document: §2.2.

[22] Charanjit S. Jutla, *Encryption Modes with Almost Free Message Integrity* **2045** (2001), 529–544, *Eurocrypt*, Lecture Notes in Computer Science, 2001. Citations in this document: §1.1, §1.1.

[23] T. Kohono, J. Viega and D.Whiting, *CWC : A High Performance Conventonal Authenticated Encryption Mode* **3017** (2004), 408–426, *Fast Software Encryption*, Lecture Notes in Computer Science, 2004. Citations in this document: §1.1.

[24] T.Krovetz and P.Rogaway, *The Software Performance of Authenticated-Encryption Modes* **6733** (2011), 306–327, *Fast Software Encryption*, Lecture Notes in Computer Science, 2011.

[25] Michael Luby, Charles Rackoff, *How to construct pseudorandom permutations from pseudorandom functions*, SIAM Journal of Computing (1988), 373–386.

[26] S. Lucks, *Two Pass Authenticated Encryption Faster than Generic Composition* **3557** (2005), 284–298, *Fast Software Encryption*, Lecture Notes in Computer Science, 2005. Citations in this document: §1.1.

[27] M. Nandi, *Two New Efficient CCA-Secure Online Ciphers: MHCBC and MCBC* **5365** (2008), 350–362, *Indocrypt*, Lecture Notes in Computer Science, 2008. Citations in this document: §1.1.

[28] M.Nandi, *A Generic Method to Extend Message Space of a Strong Pseudorandom Permutation.*, Computacin y Sistemas **12** (2009).

[29] Jacques Patarin, *The "Coefficients H" Technique* **5381** (2009), 328–345, *Selected Areas in Cryptography*, Lecture Notes in Computer Science, 2009. Citations in this document: §2.1.

[30] Bart Preneel and Hongjun Wu, *AEGIS: A Fast Authenticated Encryption Algorithm*, *Cryptology ePrint Archive: Report 2013/695.*

[31] Thomas Ristenpart and Phillip Rogaway, *How to Enrich the Message Space of a Cipher* (2007), *Fast Software Encryption*, 2007. Citations in this document: §3.

[32] P.Rogaway, *Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC* **3329** (2004), 16–31, *Asiacrypt*, Lecture Notes in Computer Science, 2004.

[33] P.Rogaway, *Nonce-based symmetric encryption* **3017** (2004), 348–359, *FSE 2004*, Lecture Notes in Computer Science, 2004.

[34] Phillip Rogaway and Haibin Zhang, *Online Ciphers from Tweakable Blockciphers* (2011), 237–249, *CT-RSA*, 2011. Citations in this document: §1.1.

[35] P.Rogaway and T.Shrimpton, *Deterministic Authenticated-Encryption : A Provable-Security Treatment of the Key-Wrap Problem* **4004** (2006), 373–390, *Advances in Cryptology - Eurocrypt*, Lecture Notes in Computer Science, 2006. Citations in this document: §1.1.

[36] Palash Sarkar, *On Authenticated Encryption Using Stream Ciphers Supporting an Initialisation Vector*, IACR Cryptology ePrint Archive (2011), 299–299. URL: http//eprint.iacr.org/2011/299.pdf. Citations in this document: §1.1.
capsulating Security Payload (ESP)

[37] J.Viega and D.McGraw, *The use of Galois/Counter Mode (GCM) in IPsec En-* (2005), *RFC 4106*, 2005. Citations in this document: §1.1.