Revisiting Security Claims of XLS and COPA

Mridul Nandi

Applied Statistics Unit
Indian Statistical Institute, Kolkata
mridul@isical.ac.in

Abstract. Ristenpart and Rogaway proposed XLS in 2007 which is a generic method to encrypt messages with incomplete last blocks. Later Andreeva et al., in 2013 proposed an authenticated encryption COPA which uses XLS while processing incomplete message blocks. Following the design of COPA, several other CAESAR candidates used the similar approach. Surprisingly in 2014, Nandi showed a three-query distinguisher against XLS which violates the security claim of XLS and puts a question mark on all schemes using XLS. However, due to the interleaved nature of encryption and decryption queries of the distinguisher, it was not clear whether the security claims of COPA remains true or not. This paper revisits XLS and COPA both in the direction of cryptanalysis and provable security. Our contribution of the paper can be summarized into following two parts:

1. **Cryptanalysis**: We describe two attacks - (i) a new distinguisher against XLS and extending this attack to obtain (ii) a forging algorithm with query complexity about $2^{n/3}$ against COPA where n is the block size of the underlying blockcipher.
2. **Security Proof**: Due to the above attacks the main claims of XLS (already known before) and COPA are wrong. So we revise the security analysis of both and show that (i) both XLS and COPA are pseudorandom function or PRF up to $2^{n/2}$ queries and (ii) COPA is integrity-secure up to $2^{n/3}$ queries (matching the query complexity of our forging algorithm).

Keywords: XLS, COPA, Pseudorandom function, Authenticated Encryption, forgery, distinguisher.

1 Introduction

Domain Completion. The notion of *domain extension* (also popularly known as modes of operations) arises in hash function [12, 23], pseudorandom function (PRF) or permutation (PRP) [15, 21], message authentication code [7, 34], authenticated encryption [20, 28], strong pseudorandom permutation (SPRP) [21] etc. The goal of a domain extension is to extend a smaller domain of an underlying primitive to a much larger domain [13]. In this paper, we study blockcipher based domain extensions which construct different variants of encryption algorithms over domain larger than $I_n := \{0,1\}^n$ where n is the block size of the underlying blockcipher. Usually, it is easy to define encryption algorithms over

$I_n^+ := \cup_{i=1}^{\infty} I_n^i$ than arbitrary bit strings $\{0,1\}^*$. We call an encryption algorithm *full-block encryption* [2, 4, 5, 18, 19, 24, 31] if its domain is I_n^+ or $I_n^{[1..a]} := \cup_{i=1}^{a} I_n^i$ for some reasonably large a. An *all-block encryption* (or simply encryption) can encrypt messages with incomplete blocks as well. In practice, it is always desired to use all-block encryptions [1, 8, 16, 17, 22, 29, 31–33]. However, full-block encryptions usually appear (as they are usually easier to define) during the process of defining all-block encryptions. A method is called **domain completion** for encryption if it *generically converts a full-block encryption into an all-block encryption.*

Examples of Generic Domain Completion. To our best knowledge, only two domain completions have been proposed for length-preserving encryption as illustrated in the Fig. 1 ((1) and (2)). The oldest method, but heuristically described, is the **Elastic blockcipher** [9] proposed by Cook, Yung and Keromytis (see (3) of Fig. 1). Later it has been more formally defined over $\{0,1\}^{[n..2n]} := \cup_{i=n}^{2n} \{0,1\}^i$ [10]. Following their paradigm (encrypt-then-mix), an efficient, neatly defined domain completion method XLS or eXtended by Latin Square is proposed by Ristenpart and Rogaway [27] ((1) of the Fig. 1 and see Algorithm 1 in section 2) which can encrypt any messages of size at least n. It requires two invocations of an n-bit blockcipher E and one invocation of a length-preserving full-block encryption $\mathcal{E}$. The second domain completion method uses hash-counter- hash paradigm [25]. In case of authenticated encryptions (supporting both confidentiality and integrity) several individual methodologies, such as tag-splitting [14]), counter-based encryption [16, 25], ciphertext stealing [30] etc. are known. Due to the attractive performance and simplicity of XLS, it is used to define an all-block authenticated encryption COPA [3] (see Fig. 2 in section 2) proposed in 2013. Following the design paradigm of COPA, several other CAESAR candidates[1] e.g., Deoxys, Joltik and KIASU, SHELL etc., used the similar approach.

Our Contribution. Recently, Nandi in Asiacrypt 2014 [26] showed that XLS is not strong pseudorandom permutation as claimed by the designers. This raises the following questions which would be answered in this paper.

– Is the attack of XLS can be extended to those encryption schemes, e.g. COPA, which uses it?

– As the security claims of XLS becomes wrong what security guarantee can be claimed?

– Do the security claims of COPA need to be revised?

The known distinguisher of XLS makes an encryption query followed by a decryption and then an encryption. In the forging security game of COPA, adversary can first make encryption queries and finally a decryption query to forge. It does not get an access of encryption queries later on which is required to attack XLS. Thus, the known attack against XLS can not be used to COPA.

1. **A new distinguisher of XLS**. We demonstrate an adaptive chosen plaintext-ciphertext distinguisher (see Algorithm $\mathcal{A}_0$ in section 3.1) against XLS mak-

[1] A competition for obtaining portfolio of secure authenticated encryptions [1].

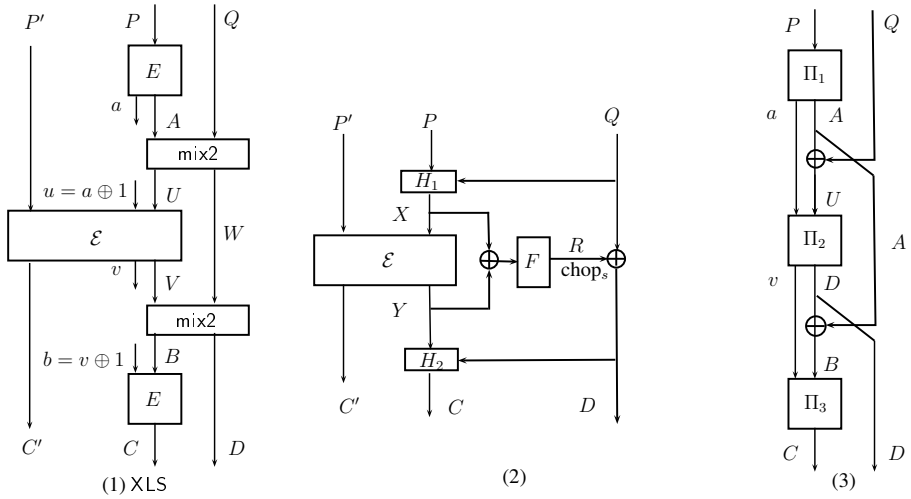

Fig. 1. $\mathcal{E}$ and E are length-preserving encryption schemes over I_n^+ and I_n respectively. $P', C' \in I_n^*$ with $|P'| = |C'|$, $X, Y, C, P \in I_n$ and $D, Q, A, B, U, V \in \{0,1\}^s$, $1 \leq s < n$. (1) **XLS**: mix2 : $\cup_{s=1}^{n-1}\{0,1\}^{2s} \rightarrow \cup_{s=1}^{n-1}\{0,1\}^{2s}$ defined as mix2$(AB) = (A \oplus (A \oplus B)^{\lll 1}, B \oplus (A \oplus B)^{\lll 1})$ where $\lll a$ denotes a-bit left rotation. (2) **Hash-Counter-Hash Paradigm**: H_1 and H_2 are universal hash functions which could be instantiated by field multiplication or AES [11] with four rounds and F is a weak pseudorandom function. (3) **Elastic blockcipher**: It encrypts only over $\{0,1\}^{[n..2n]}$.

ing two encryption queries followed by one decryption query. The basic principle of the new attack of XLS remains same. So, this new attack becomes more suitable for applying to forging game of COPA.

2. **A Forging Algorithm of COPA**. The next immediate step is to see the implication of insecurity of XLS to COPA or others in which it has been applied. In this paper, we extend the above distinguishing attack to make a forging algorithm $\mathcal{A}_1$ (see in section 3.2) on a general design paradigm of COPA and hence to specific designs such as AES-COPA, Deoxys, Joltik, KIASU etc. The algorithm SHELL uses a masking before it applies XLS. However, the same algorithm can be carried out even with the presence of masking. The algorithm makes $2^{n/3}$ encryption queries and make one forging attempt with success probability about $1/4$. This violates the security claim proved in Asiacrypt 2013 [3].[2]
3. **XLS has $n/2$-bit PRF security and COPA has $n/3$-bit integrity security**. The above two negative results disprove the claims for XLS and COPA. So we need to analyze how much security guarantee can be achieved. In Theorem 3, we show that XLS (in fact, a more general version of XLS called GXLS) is a pseudorandom function (PRF). Using this property, the privacy

[2] However, our algorithm is reduced to solving generalized birthday attack [35] for three lists. So we do not know any algorithm which can run in time-complexity significantly less than $2^{n/2}$. But the main security claim of [3] was proved even for unbounded adversary and this is why the result become wrong.

of COPA can be easily shown. In Theorem 4, the upper bound of forging advantage is shown to be about $q^3/2^n$ where q is the number of queries. This also shows the tightness of query complexity of forging security of COPA.

2 Known Results of XLS and COPA

2.1 Description of XLS and COPA

Input: $M \in \{0,1\}^{\geq n}$
, **Output**: $Z \in \{0,1\}^{\geq n}$, $|Z| = |M|$, **Key**: keys of E and $\mathcal{E}$.

Algorithm XLS(M)

let $|M| = nk + s$, $0 \leq s < n$, $k \geq 1$
if $s = 0$ **then**
  return $Z = \mathcal{E}(M)$.
else
  parse $M = (P', P, Q)$, $P' \in \{0,1\}^{n(k-1)}$, $P \in \{0,1\}^n$, $Q \in \{0,1\}^s$
  $a\|A \leftarrow E(P)$, $a \in \{0,1\}^{n-s}$, $A \in \{0,1\}^s$.
  $u \leftarrow a \oplus 1$, $(U, W) \leftarrow \mathsf{mix2}(A, Q)$.
  $(C', v\|V) \leftarrow \mathcal{E}(P', u\|U)$, $v \in \{0,1\}^{n-s}$, $V \in \{0,1\}^s$, $|C'| = |P'|$.
  $b \leftarrow v \oplus 1$, $(B, D) \leftarrow \mathsf{mix2}(V, W)$.
  $C \leftarrow E(b\|B)$.
  return $Z = (C', C, D)$.

Algorithm 1: The decryption algorithm $\mathsf{XLS}^{-1}(Z)$ is exactly same as the encryption algorithm except that we replace E and $\mathcal{E}$ by E^{-1} and $\mathcal{E}^{-1}$ respectively. Here, $1 = 0^{n-s-1}1$. XOR-ing with 1 is necessary to handle variable length input.

XLS: Let $\mathcal{E}$ be a length-preserving encryption on I_n^+ and E be an n-bit blockcipher. For $|A| = |B| = s$, $1 \leq s \leq n-1$, we define an efficiently computable linear involution (a permutation with self-inverse): $\mathsf{mix2}(A, B) = (\mathsf{mix2}_L := A \oplus (A \oplus B)^{\lll 1}$, $\mathsf{mix2}_R := B \oplus (A \oplus B)^{\lll 1})$ where $X^{\lll a}$ (or $X^{\ggg a}$) denotes a-bit left rotation (or right rotation respectively). The algorithmic description of XLS is described in Algorithm 1.

COPA: It is an all-block authenticated encryption expanding n-bits in ciphertext. It first defines a full-block encryption $\mathcal{F}_{\mathsf{COPA}}$. As the details of the construction of $\mathcal{F}_{\mathsf{COPA}}$ is not required we skip the actual definition which can be found in [3]. The similar paradigm of COPA is also used for other authenticated encryptions. We describe it as a generic domain competition of an authenticated encryption for a message $M[1..d] := (M[1], \ldots, M[d])$ with $|M[d]| = s$ as follows:

$$\overline{\mathcal{F}}(M) = \begin{cases} C[1..d-1] \parallel S & \text{if } s = 0 \text{ (last block is complete);} \\ C[1..d-1] \parallel \mathsf{XLS}(M[d]\|S) & \text{if } s > 0 \text{ (last block is incomplete);} \end{cases}$$

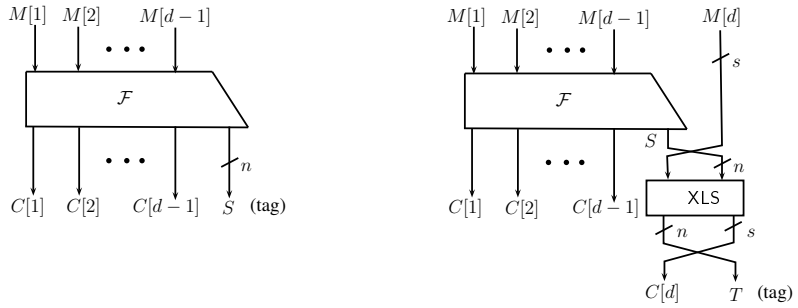

Fig. 2. Let $\mathcal{F}$ be a full-block encryption. In L.H.S. we simply apply it for full-block messages M. R.H.S. describes the ciphertext computation for messages with incomplete last block $M[d]$ where $1 \le s = |M[d]| < n$. XLS is defined over $\{0,1\}^{[n+1..2n-1]}$.

where $\mathcal{F}(M[1..d-1]) := C[1..d-1] \parallel S$. Here, the XLS gets only inputs of sizes in between $n+1$ and $2n-1$. The pictorial illustration of the above domain competition is given in the Fig **??**.

2.2 Security Definitions

We first briefly recall security definitions related to encryption and authenticated encryption schemes. We define the distinguishing advantage of A distinguishing f_K from g_L is defined as

$$\mathbf{Adv}_f^g(A) := |\mathsf{Pr}_K[A^{f_K} = 1] - \mathsf{Pr}_L[A^{g_L} = 1]|$$

where the two probabilities are computed under randomness of keys K and L respectively, and random coin of A. Similarly one can define advantages for two oracles such as f_K and its inverse f_K^{-1} when f_K is a permutation. An ideal random function (or permutation) \$ (or Π) returns random strings R (keeping permutation property) for every fresh query. Whereas an ideal random source returns uniform and independent string for every query, even repeated. We write $\mathbf{Adv}_f^{\mathrm{prf}}(A) := \mathbf{Adv}_f^{\$}(A)$, $\mathbf{Adv}_f^{\mathrm{prp}}(A) := \mathbf{Adv}_f^{\Pi}(A)$ and $\mathbf{Adv}_f^{\mathrm{sprp}}(A) := \mathbf{Adv}_{(f,f^{-1})}^{(\Pi,\Pi^{-1})}(A)$.

Online Distinguishing Advantage. Let $\$_{ol}$ and Π_{ol} denote online random function and permutation respectively which behave like random keeping the "online property" – the i^{th} block (not the last block) of ciphertext only depends on the first i blocks of plaintext, not on the subsequent blocks. We similarly define $\mathbf{Adv}_f^{\mathrm{olprf}}(A)$, $\mathbf{Adv}_f^{\mathrm{olprp}}(A)$ and $\mathbf{Adv}_f^{\mathrm{olsprp}}(A)$. We define the maximum xxx-advantage $\mathbf{Adv}_f^{\mathrm{xxx}}(t,q,\sigma,\ell) = \max_A \mathbf{Adv}_f^{\mathrm{xxx}}(A)$ where the maximum is taken over all algorithms A which run in time t, make queries at most q with the total number of blocks at most σ and the number of blocks in the longest query is ℓ. For unbounded time adversaries, $\mathbf{Adv}_f^{\mathrm{xxx}}(q,\sigma,\ell) = \mathbf{Adv}_f^{\mathrm{xxx}}(\infty,q,\sigma,\ell)$.

Integrity Security of an Authenticated Encryption. An authenticated encryption is called secure if it is pseudorandom function as well as unforgeable,

i.e., the authenticity advantage, defined below is negligible.

$\mathbf{Adv}^{\mathrm{auth}}_{\mathsf{AE}}(A) := \mathsf{Pr}_k[A^{\mathsf{AE}_k} \to (C),\ C \text{ is not obtained by } \mathsf{AE} \text{ queries}, \mathsf{AE}_k^{-1}(C) \neq \perp]$.

2.3 Known Distinguisher of XLS

Here we briefly describe the known results for XLS and COPA. In [27], it has been proved that XLS is strong pseudorandom permutation (SPRP). Later, in Asiacrypt 2014 [26] Nandi showed a three-query CPCA distinguisher against XLS which violates the designer's claim. Now, we briefly state the distinguisher for XLS. It first makes an encryption query (P_1, Q_1) and obtains a response (C_1, D_1) where $|P_1| = |C_1| = n$ and $|Q_1| = |D_1| = n-1$. Then, it makes a decryption query $(C_2 := C_1, D_2 := D_1 \oplus 1)$ and obtains a response (P_2, Q_2). Finally, it makes an encryption query $(P_3 := P_2, Q_3 := Q_2 \oplus 1 \oplus (Q_1 \oplus Q_2 \oplus 1)^{\lll 2})$ and obtains a response (C_3, D_3). In [26], it was shown that if the distinguisher is interacting with XLS, $D_3 = Q_1 \oplus Q_3 \oplus D_1$ holds with probability $1/2$ whereas this clearly does not hold almost with probability $1 - 2^{-n+1}$ for a random permutation. This clearly leads a distinguishing attack.

2.4 Security Claims of COPA

In Asiacrypt 2013 [3], COPA has been proposed which uses XLS for processing arbitrary length messages. Moreover, it has been shown that COPA is privacy and authenticity secure. As the security claim of XLS becomes wrong, the security claim of COPA is required to be revised. Note that the above attack has two levels of adaptiveness. So the attack can not be deployed immediately into an attack of COPA. In a privacy attack only encryption queries can be made. Whereas in the forging attack, a forger is allowed to make first encryption queries and then a decryption query to forge. So the security claims of COPA remains open as the power of the distinguisher is more than a forging algorithm. Now we state main results for COPA claimed in Asiacrypt 2013. Here, the building block $\mathcal{F}_{\mathsf{COPA}}$ and XLS are based on an n-bit blockcipher E.

Theorem 2 of [3].

$$\mathbf{Adv}^{\mathrm{oprf}}_{\mathsf{COPA}}(t, q, \sigma, \ell) \leq \mathbf{Adv}^{\mathrm{sprp}}_{E}(t', 4(\sigma+q)) + \frac{39(\sigma+q)^2}{2^n} + \frac{(\ell+2)(q-1)^2}{2^n}$$

where σ and ℓ are the total and longest number of blocks among all q queries and t denotes time complexity. Here $t' \approx t$.

Theorem 3 of [3].

$$\mathbf{Adv}^{\mathrm{auth}}_{\mathsf{COPA}}(t, q, \sigma, \ell) \leq \mathbf{Adv}^{\mathrm{sprp}}_{E}(t', 4(\sigma+q)) + \frac{39(\sigma+q)^2}{2^n} + \frac{(\ell+2)(q-1)^2}{2^n} + \frac{2q}{2^n}.$$

When we use an ideal random permutation E and consider all unbounded adversaries (i.e., $t = \infty$) then $\mathbf{Adv}^{\mathrm{sprp}}_{E}(\infty, \sigma') = 0$ for all σ'. Hence the above

theorem says that the COPA encryption algorithm $\overline{\mathcal{F}}_{\mathsf{COPA}}$ is privacy and authenticity secure as long as $\sigma \ll 2^{n/2}$ and E is replaced by a random permutation Π. Here we assume the above theorems for $\mathcal{F}_{\mathsf{COPA}}$ over full block messages as it does not involve XLS. However, we need to re-evaluate for arbitrary length encryption $\overline{\mathcal{F}}_{\mathsf{COPA}}$.

3 Cryptanalysis of XLS and COPA

3.1 A Less Adaptive Distinguisher for XLS

We first describe a less adaptive distinguisher for XLS. Later we use the basic idea of this attack of XLS to obtain a forging algorithm for COPA violating the Theorem 3 of [3] as mentioned above. Now we describe an adversary $\mathcal{A}_0$ (see Algorithm below and illustration in Fig. 3.1) which makes four queries to XLS and its inverse having advantage about 1/2.

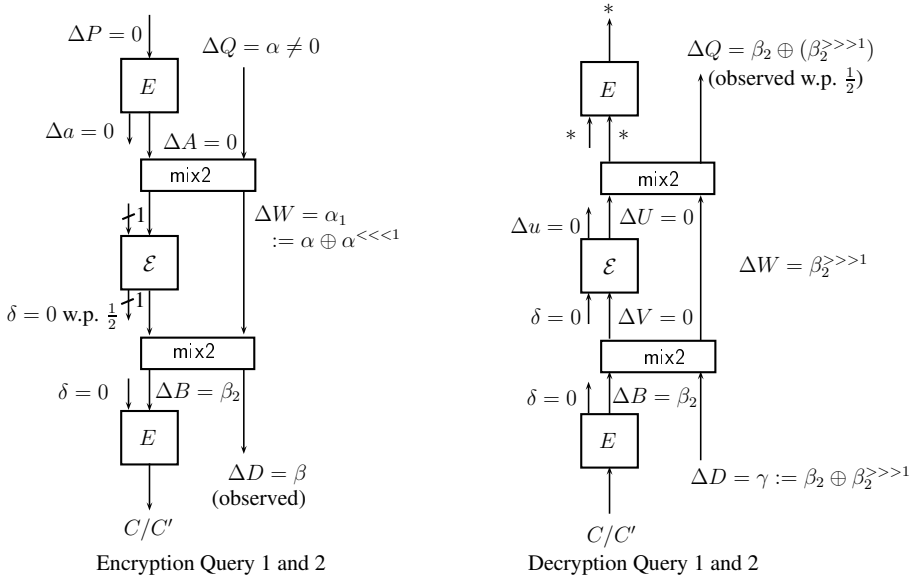

Fig. 3. Illustration of differences present in the attack: Here $|Q| = |D| = n - 1$. We observe C and C' in the first block, and β difference in second blocks of ciphertexts. From the definition of mix2, $\beta_2 = \beta \oplus (\beta \oplus \alpha_1)^{\gg 1} = \alpha \oplus \beta \oplus (\alpha \oplus \beta)^{\gg 1}$. So if we set γ difference in second ciphertext blocks of decryption queries such that inputs of $\mathcal{E}^{-1}$ collide whenever $\delta = 0$ (which can happen with probability 1/2 as δ is a single bit). So we have γ difference in the second plaintext blocks of two decryption queries which would be clearly a test for the distinguisher.

Distinguisher $\mathcal{A}_0$ for XLS with message sizes $2n-1$ (so $s=n-1$).

1. **query-1**. It makes an encryption query $(P,Q)\in\{0,1\}^n\times\{0,1\}^{n-1}$.
2. Let $(C,D)\in\{0,1\}^n\times\{0,1\}^{n-1}$ be the response.
3. Fix a non-zero bit string α of size $n-1$.
4. **query-2**. Encryption query $(P,Q':=Q\oplus\alpha)$ and obtains response (C',D').
5. Let $\beta=D\oplus D'$ and set $\gamma=\alpha\oplus\beta\oplus(\alpha\oplus\beta)^{\gg 2}$.
6. **query-3**. It makes a decryption query (C,D_1).
7. Let (P_1,Q_1) be the response.
8. **query-4**. Decryption query $(C',D_1':=D_1\oplus\gamma)$ and obtains response (P_1',Q_1').
9. **if** $Q_1'\oplus Q_1=\gamma$ **returns** 1, **else** 0.

Theorem 1. *For $\mathcal{A}_0$ (as defined above) we have* $\mathbf{Adv}^{\mathrm{sprp}}_{\mathsf{XLS}}(\mathcal{A}_0)\approx\frac{1}{2}-\frac{1}{2^{n-1}}$.

Proof. We first note that $\mathsf{mix2}$ is a linear function and so

$$\mathsf{mix2}(X\oplus\Delta_1,Y\oplus\Delta_2)\oplus\mathsf{mix2}(X,Y)=\mathsf{mix2}(\Delta_1,\Delta_2).$$

In other words, the difference of outputs of $\mathsf{mix2}$ can be computed just form the difference of inputs. In Fig. 3.1, we illustrate differences of internal variables in the computation of query-1, 2 (in left hand side of the figure) and query-3, 4 (right hand side of the figure). By $*$ we mean the difference is not required to be known. Now we explain why the differences are observed in the four queries.

1. Note that in the first two encryption queries $\Delta W:=\alpha_1=\mathsf{mix2}_R(0,\alpha)=\alpha\oplus\alpha^{\lll 1}$ since there is zero difference in outputs of the first call of E.
2. Let $\Delta B:=\beta_2$ be the differences of the first $n-1$ bits of second call of $\mathsf{mix2}$ in the first two encryption queries. Since $\mathsf{mix2}$ is an involution, we can solve β_2 easily from the equation $\alpha_1=\mathsf{mix2}_R(\beta_2,\beta)=(\beta_2\oplus\beta)^{\lll 1}\oplus\beta$. So,
$$\begin{aligned}\beta_2&=(\alpha_1\oplus\beta)^{\ggg 1}\oplus\beta\\&=(\alpha\oplus\beta)\oplus(\alpha\oplus\beta)^{\ggg 1}.\end{aligned}$$
3. Let $\delta=\Delta b$ which is a single bit. Thus, $\mathsf{Pr}[\delta=0]=1/2$.
4. In the decryption queries we give same pair of first ciphertext blocks, namely C and C', and so we would get β_2 difference after we invoke first E^{-1} call of decryption queries.
5. As the difference of the second ciphertext block is chosen as $\gamma:=\beta_2\oplus\beta_2^{\ggg 1}$, we have $\mathsf{mix2}(\beta_2,\gamma)=(0^{n-1},\beta_2^{\ggg 1})$. Given that $\delta=0$, we have a collision on the first $\mathcal{E}^{-1}$ calls for decryption queries. So we have $\Delta U=0$ in the decryption queries.
6. Finally, the difference in second plaintext blocks of decryption query is $\mathsf{mix2}_R(0,\mathsf{mix2}_R(\beta_2,\gamma))=\mathsf{mix2}_R(0,\beta_2^{\ggg 1})=\gamma$.

Thus, with probability $1/2$, $Q_1\oplus Q_1'=\gamma$, i.e., $\mathsf{Pr}[\mathcal{A}_0^{\mathsf{XLS},\mathsf{XLS}^{-1}}=1]=1/2$. When $\mathcal{A}_0$ is interacting with random permutation P and its inverse P^{-1}, $\mathsf{Pr}[\mathcal{A}_0^{\mathsf{P},\mathsf{P}^{-1}}=1]\approx 2^{1-n}$. So we are done. □

Remark 1 (We need three queries instead of four). One can clearly drop the third query (i.e., the first decryption query) by setting $D_1 = D$ and so the response $(P_1, Q_1) = (P, Q)$ is known from the first encryption query. We exploit this while extend this attack to COPA.

3.2 Forging Algorithm of COPA

In this section, we demonstrate a forgery algorithm against COPA and all other similar candidates of authenticated encryptions (e.g., Deoxys, Joltik, KIASU, SHELL[3]). We do not need any weakness on $\mathcal{F}_{\mathsf{COPA}}$ and hence it can be any strong authenticated encryption or random online injective function. The algorithm makes $q \approx 2^{n/3}$ queries.

Forgery Algorithm $\mathcal{A}_1$ for Authenticated Encryption $\mathcal{F}$.

1. queries $M_i \in \{0,1\}^n$ and response $(C_i, t'_i \| Q_i)$ where $|t'_i| = 1$, $1 \le i \le q$.
2. Find b (assume $b = 0$), $|I| = |\{i : t'_i = b\}| \ge q/2$. $I = I_1 \sqcup I_2$, $|I_1| = |I_2|$.
3. queries (M_i, m), $i \in I$, $m \in \{0,1\}^{n-1}$ and responses $((C_i, D_i), T_i)$.
4. Find $i \in I_1, j \in I_2, k \in I$ s.t.

$$Q_k = (D_i + Q_i)^{\lll 2} + Q_j + (D_j + Q_j)^{\lll 2},$$

   otherwise **abort**.
5. Return forgery query (C_k, D^*, T_j) where

$$D^* = (D_i + Q_i + Q_j) + (D_i + Q_i + D_j + Q_j)^{\lll 2}.$$

3.3 Analysis of Success Probability of Forgery

For each index i, $1 \le i \le q$, we associate three lists of $(n-1)$ bit strings (1) Q_i, (2) $\alpha_i := (D_i + Q_i)^{\lll 2}$ and and (3) $\beta_i = Q_j + (Q_j + D_j)^{\lll 2}$. In the step 4, we try to find $i \in I_1, j \in I_2, k \in I$ such that $Q_k = \alpha_i + \beta_j$. This is actually a generalized birthday problem to find such a solution and we do not know so far the exact time complexity for the generalized birthday problem for three or more lists. However, here we analyze it in terms of query complexity and so the time complexity could be more, e.g. about $2^{2n/3}$. The main reason is to choose I_1 and I_2 disjoint to make α_i, β_j and Q_k independent. If $|I| = 2^{n/3+1}$ then there are about 2^{n+1} many possible values of $Q_k + \alpha_i + \beta_j$ and we expect one such triple, for which it takes value zero. So we set $q = 2^{\frac{n}{3}+2}$ and so by pigeonhole principle, $|I| \ge 2^{n/3+1}$. We first note that α_i, β_j and Q_k's follow uniform and independent distributions (otherwise we would be able to mount distinguishing attack on this mode). Moreover, we can assume that $\mathcal{F}$ is pseudorandom function (or random function) on complete block messages, otherwise we also have a distinguishing

[3] Truly speaking SHELL uses a masking before it applies XLS. However, it can be easily seen that the same attacks works for any linear key masking.

attack. So Q_i's are independent and uniformly distributed. As we have chosen α_i and β_j from disjoint sets I_1 and I_2, Q_k, α_i and β_k are independent. Armed with these basic observations, we have our analysis of the forging algorithm.

Theorem 2. *Given that $\mathcal{A}_1$ does not abort, it forges with success probability $1/4$. Moreover, when $q = 4 \times 2^{n/3}$, on the average we do not expect to abort.*

Proof. We have already argued why we do not expect to abort. Now we show that when it does not abort, i.e., $\alpha_i + \beta_j = Q_k$ the forging succeeds with probability at least $1/4$. We recall the main properties from algorithm $\mathcal{A}_0$. We have shown that if (C_i, D_i) and (C_j, D_j) are responses of (P, Q_i) and (P, Q_j) respectively then the decryption of $(C_j, D_{i,j})$ is (P', Q') with probability $1/2$ where

1. $D_{i,j} = D_i + Q_i + Q_j + (D_i + Q_i + D_j + Q_j)^{\lll 2}$ and
2. $Q' = D_{i,j} + (Dj_i + Q_i) = (D_i + Q_i)^{\lll 2} + Q_j + (Q_j + D_j)^{\lll 2}$.

In this forging algorithm, we now have $Q' = Q_k$. So now we try to decrypt (C_k, D^*, T_j). Here T_j plays role of C_j and so when we apply $\mathsf{XLS}^{-1}(T_j, D^*)$ we obtain $(m^*\|t^*), Q_k)$ for some $m^*\|t^* \in \{0,1\}^n$. Now $t^* = 0$ with probability $1/2$ and so when we run the verification algorithm for $(C_k, 0\|Q_k)$, it passes and returns the message M_k. So we forge a message of the form (M_k, m^*) for some M_k controlled by adversary and unknown m^* with success probability about $1/4$. □

The abort in step 4 can be checked without making any forging query. However, the bit t^* is 0 can not be verified without making verification query. So if we repeat the forging algorithm k times we can increase the success probability of forging almost to $1/2$ for a single forging attempt.

Remark 2 (reason why it violates Theorem 3 (integrity security claim) of COPA). So far we do know any algorithm with time complexity less than $2^{n/2}$ which finds such i, j and k. However, in terms of query complexity, our algorithm requires less query complexity, i.e. about $2^{n/3}$ than what claimed in [3], i.e., $2^{n/2}$ even for an unbounded adversary. Note that our attack works for all types of underlying blockcipher even for ideal random permutation.

4 XLS is a Pseudorandom function

4.1 A General Form of Encrypt-then-Mix Encryption

We extend the definition of this permutation to the domain $\cup_{s=1}^{n-1}\{0,1\}^{n+s}$ as follows. $\mathsf{mix2}'(X_1\|X_1, Y) = (\mathsf{mix2}'_L, \mathsf{mix2}'_R)$ where

$$\mathsf{mix2}'_L(X_1\|X_2, Y) = (X_1 \oplus 1)\|\mathsf{mix2}_L(X_2, Y), \text{ and } \mathsf{mix2}'_R(X_1\|X_2, Y) = \mathsf{mix2}_R(X_2, Y).$$

Here, 1 is the bit string of size $n - s$ whose last bit is 1 and all other bits are zero. So $\mathsf{mix2}'$ is an involution from $\{0,1\}^{n+s}$ to itself, $1 \le s \le n-1$.

Input: $(P, Q) \in \{0,1\}^n \times \{0,1\}^s$, $1 \leq s \leq n-1$.
Output: $(C, D) \in \{0,1\}^n \times \{0,1\}^s$.
Key: keys of permutations Π_1, Π_2 and Π_3.

Algorithm GXLS(P, Q)

1 $A \leftarrow \Pi_1(P)$.
2 $(U, W) \leftarrow \mathsf{mix}_1(A, Q)$. \\ the functions mix_1 and mix_2 are linear permutations on $\{0,1\}^{n+s}$.
3 $V \leftarrow \Pi_2(U)$.
4 $(B, D) \leftarrow \mathsf{mix}_2(V, W)$.
5 $C \leftarrow \Pi_3(B)$.
6 **return** (C, D).

Algorithm 2: The encryption algorithm GXLS.

A general version of XLS (we call it GXLS) in which $\mathsf{mix2}'$ function is replaced by two invertible linear functions mix_1 and mix_2. In order to make the encryption scheme invertible, we need mix functions to be invertible. Another generalization is to consider three independent random permutation instead of two. Whenever we restrict $\mathcal{E}$ to be an n-bit blockcipher, the GXLS is defined over the domain $\cup_{i=n+1}^{2n-1}\{0,1\}^i$. It can also be seen as three iterations of encrypt (for n bits) and mix functions except that the last round which does not have mixing. Note that the elastic blockcipher also falls in this paradigm where the number of rounds can vary. The above algorithm is also denoted as $\mathsf{GXLS}^{\Pi_1,\Pi_2,\Pi_3}[\mathsf{mix}_1, \mathsf{mix}_2]$. Note that decryption algorithm is $\mathsf{GXLS}^{\Pi_1^{-1},\Pi_2^{-1},\Pi_3^{-1}}[\mathsf{mix}_1^{-1}, \mathsf{mix}_2^{-1}]$. With this notation, XLS is nothing but $\mathsf{GXLS}^{E,\mathcal{E},E}[\mathsf{mix2}', \mathsf{mix2}']$. Similarly, three round elastic blockcipher can be described. When the mix functions are understood from the context we simply write $\mathsf{GXLS}^{\Pi_1,\Pi_2,\Pi_3}$. Note that we can view it as an oracle algorithm which interacts with three permutations or any systems (could be stateful) Π_1, Π_2 and Π_3.

4.2 Generalized XLS is a Pseudorandom Function

In this section we prove that $\mathsf{XLS}^{E,\mathcal{E},E}$ is a pseudorandom function whenever E and $\mathcal{E}$ are pseudorandom functions used to compute XLS. In the security we consider hybrid constructions by replacing these by ideal objects. Moreover, we generalize XLS construction by considering general mixing functions. The figure for a more general construction and the elastic blockcipher are illustrated in Appendix B. We actually show that GXLS is pseudorandom function provided that mixing functions satisfy certain conditions which are actually satisfied by mix2.

Some Properties of the Mixing Function mix2 For any length-preserving permutation π on $\cup_{s=1}^{n-1}\{0,1\}^{n+s}$, we denote $\pi(z) = \pi_L(z)\|\pi_R(z)$ where $\pi_L(z) \in \{0,1\}^n$ and $\pi_R(z) \in \{0,1\}^s$. Now we define some properties of a length-preserving

permutation $\pi = (\pi_L, \pi_R)$ on $\cup_{s=1}^{n-1}\{0,1\}^{n+s}$. A function $f : D \to R$ is called regular if $U \xleftarrow{\$} D$ implies $f(U) \xleftarrow{\$} R$. In other words, for all $y \in R$, $\mathsf{Pr}[f(U) = y] = |R|^{-1}$ where U is uniformly chosen from D. A function f is called ϵ-regular if for all $y \in R$, $\mathsf{Pr}[f(U) = y] \le \epsilon$. So a $|R|^{-1}$-regular function is nothing but a regular function.

1. **LR-regular**: For all $B \in \{0,1\}^s$, the function $\pi_R(\cdot, B) : \{0,1\}^n \to \{0,1\}^s$ is regular.
2. **RL-injection**: For all $x \in \{0,1\}^n$ and $B \neq B'$, we have $\pi_L(x, B) \neq \pi_L(x, B')$.
3. **LL-bijection**: For all $x \in \{0,1\}^s$ and $A \neq A'$, we have $\pi_L(A, x) \neq \pi_L(A', x)$.
4. **LL-ϵ-regular**: For all $B \in \{0,1\}^s$ and all $y \in \{0,1\}^n$, $\mathsf{Pr}[\pi_L(X, B) = y; X \xleftarrow{\$} \{0,1\}^n] \le \epsilon$. If $\epsilon = 2^{-n}$ then it is equivalent to LL-bijection.

Lemma 1. *The* $\mathsf{mix2}'$ *function is LR-regular, RL-injection and LL-*2^{1-n}*-regular.*

Proof. (1) **LR-regular**: $\mathsf{mix2}_R(a, x, B) = x^{\lll} \oplus (B^{\lll} \oplus B)$. So clearly, the function mapping x to $\mathsf{mix2}_R(a, x, B)$ is a permutation for all a and so mapping (a, x) to $\mathsf{mix2}_R(a, x, B)$ is a regular function.

(2) **RL-injection**: Let $\mathsf{mix2}'_L(x, B) = \mathsf{mix2}'_L(x, B')$ for any B, B' and $x \in \{0,1\}^n$. Suppose, $|B| = |B'|$. Then, clearly, $B = B'$. So assume that $s = |B| < |B'|$. Then, the $(n-s)^{\text{th}}$ bit of $\mathsf{mix2}'_L(x, B)$ is not same as that of $\mathsf{mix2}'_L(x, B')$. So we arrive contradiction.

(3) **LL-2^{1-n}-regular**: Let $X = (X_1, X_2) \xleftarrow{\$} \{0,1\}^{n-s} \times \{0,1\}^n$. Now for all $B \in \{0,1\}^s$ and all $y = (y_1, y_2) \in \{0,1\}^{n-s} \times \{0,1\}^s$, $\mathsf{Pr}[\pi_L(X, B) = y] = \mathsf{Pr}[X_1 = y_1 \oplus 1, X_2 \oplus X_2^{\lll} = B^{\lll} \oplus y_2]$. As X_1 and X_2 are independent and the mapping $x \mapsto x \oplus x^{\lll}$ is a two-to-one function. Thus, it is a LL-2^{1-n}-bijection. □

Main Security Claim of GXLS

To simplify the notation we write $\mathbf{Adv}^{\text{prf}}_{\mathsf{GXLS}}(q)$ to mean $\mathbf{Adv}^{\text{prf}}_{\mathsf{GXLS}}(t, q, 2q, 2)$ for some t.

Theorem 3. *Suppose* mix_1 *is LL-*ϵ*-regular, LR-regular, RL-injection, and* mix_2 *is LL-*ϵ*-regular, LR-regular then*

$$\mathbf{Adv}^{\text{prf}}_{\mathsf{GXLS}}(q) \le \mathbf{Adv}^{\text{prf}}_{E}(2q) + \mathbf{Adv}^{\text{prf}}_{\mathcal{E}}(q) + 2.5q(q-1)\epsilon.$$

Proof. For notational simplicity, we write $\Pi_1 \cdot \Pi_2 \cdot \Pi_3$ to denote $\mathsf{GXLS}^{\Pi_1, \Pi_2, \Pi_3}[\mathsf{mix}_1, \mathsf{mix}_2]$. We first define different games or interactive systems which interacts with distinguishers making no repetition queries. Let A be a chosen plaintext adversary which makes exactly q distinct queries out of which q_1 many queries have incomplete last block (i.e., the query size is not multiple of n).

1. Let G_0 be the real GXLS based on keyed permutations E and $\mathcal{E}$ and the mixing functions, i.e., $G_0 = E \cdot \mathcal{E} \cdot E$.

2. We consider an intermediate system G_1 in which E and $\mathcal{E}$ are replaced by two independent uniform random functions $\mathtt{R}_1$ and $\mathtt{R}_2$. In notation, $G_1 := \mathtt{R}_1 \cdot \mathtt{R}_2 \cdot \mathtt{R}_1$. By using standard hybrid argument we have

$$|\mathsf{Pr}[A^{G_0} = 1] - \mathsf{Pr}[A^{G_1} = 1]| \leq \mathbf{Adv}^{\mathrm{prf}}_{E}(2q_1) + \mathbf{Adv}^{\mathrm{prf}}_{\mathcal{E}}(q). \quad (1)$$

3. We also consider another intermediate system $G_2 := \mathtt{R}_1 \cdot \mathtt{S}_1 \cdot \mathtt{S}_2$ where $\mathtt{S}_i$'s denote the uniform random sources which return uniform string for every query. The equivalent pseudocode descriptions of G_1 and G_2 are given in Algorithm 3 (see Appendix C). Note that these are identical except that the game G_1 executes boxed statements whereas G_2 does not. Whenever the boxed statements are getting executed, we set BAD to be 1. In other words, G_1 and G_2 are identical until bad.

$$|\mathsf{Pr}[A^{G_1} = 1] - \mathsf{Pr}[A^{G_2} = 1]| \leq \mathsf{Pr}[\mathsf{BAD} = 1 \text{ in the interaction } A^{G_2}]. \quad (2)$$

   Note that a uniform random function behaves like a uniform random source if it gets all distinct inputs. So the bad event actually captures the collisions among inputs of $\mathtt{R}_1$ and $\mathtt{R}_2$'s. More formally,
   (a) input collisions of $\mathtt{R}_2$,
   (b) input collisions of the second invocation of $\mathtt{R}_2$,
   (c) input collision between the first and second invocations of $\mathtt{R}_1$.
4. Finally, we define $G_3 := \mathtt{S}^{(2)}$ which returns $2n$ bits uniform random strings.

Lemma 2. *If mix_2 is LR-regular then the Game G_2 is equivalent to G_3.*

Proof. Note that for each query, G_2 returns $(\mathtt{S}_2(B), \mathsf{mix}_{2\,R}(\mathtt{S}_1(U), W))$, Since mix_2 is LR-regular, by the randomness of output of random sources, output of G_2 is uniformly distributed over $\{0,1\}^{2n}$. So G_2 is equivalent to a $2n$-bit uniform random source G_3. □

Finally, we state a Lemma which bounds the probability of BAD event. We give the proof of the Lemma later in the section.

Lemma 3. *Suppose mix_1 is LL-ϵ-regular, RL-injection, and mix_2 is LL-ϵ-regular, LR-regular then we have $\mathsf{Pr}[\mathsf{BAD} = 1$ in the interaction $A^{G_2}] \leq 2.5q(q-1)\epsilon$.*

Assuming this, we have

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{GXLS}}(q) = |\mathsf{Pr}[A^{G_0} = 1] - \mathsf{Pr}[A^{G_3} = 1]| \leq \mathbf{Adv}^{\mathrm{prf}}_{E}(2q) + \mathbf{Adv}^{\mathrm{prf}}_{\mathcal{E}}(q) + 2.5q(q-1)\epsilon.$$

This proves the theorem. □

Corollary 1. *XLS and elastic blockcipher with three rounds are pseudorandom function.*

Proof. We have already seen that mix2 satisfies the given conditions of the above theorem with $\epsilon = 2^{-n+1}$ (as shown in lemma 1. So XLS is a pseudorandom function. Similarly, one can show that the mixing function corresponding to the elastic blockcipher satisfy the given conditions and hence it is also PRF. □

Remark 3. The three round elastic blockcipher with appropriate xor-ing with 1 gives a pseudorandom function. Given any such construction, we only need to check whether mixing functions satisfies those properties or not to claim pseudorandom security.

4.3 Proof of Lemma 3

Deterministic Adversary. By fixing a random coin of a probabilistic adversary A, one can find a deterministic unbounded adversary $\mathcal{A}$ such that

$$\mathsf{Pr}[\mathsf{BAD} = 1 \text{ in the interaction } A^{G_2}] \leq \mathsf{Pr}[\mathsf{BAD} = 1 \text{ in the interaction } \mathcal{A}^{G_2}].$$

Moreover, a system which behaves like a uniform random function against unbounded deterministic adversary then so does against any probabilistic adversary. So throughout the proof we assume that the adversary A is deterministic but computationally unbounded (which is as powerful as a probabilistic algorithm). Without loss of generality, we assume that A makes exactly q queries and all queries are distinct.

We state a simple lemma, called masking lemma, which is useful in general.

Lemma 4 (masking lemma). *Suppose a length-preserving permutation π is LR-regular and let Y be an s-bit random variable. Then,*

$$\forall X \xleftarrow{\$} \{0,1\}^n,\ X \perp Y \quad \Rightarrow \quad \pi_R(X,Y) \perp Y.$$

Proof. As π is LR-regular and $X \perp Y$, $\pi_R(X,Y)$ follows uniform distribution on $\{0,1\}^s$. Now we compute the joint distribution of $(\pi_R(X,Y),Y)$. For any $y,z \in \{0,1\}^s$, $\mathsf{Pr}[\pi_R(X,Y) = z, Y = y] = \mathsf{Pr}[Y = y] \times \mathsf{Pr}[\pi_R(X,y) = z]$. As π is LR-regular $\mathsf{Pr}[\pi_R(X,y) = z] = 2^{-s}$ and hence $\mathsf{Pr}[\pi_R(X,Y) = z, Y = y] = \mathsf{Pr}[\pi_R(X,y) = z] \times \mathsf{Pr}[Y = y]$. □

Notation. We first identify the source random variables (independent and uniformly distributed) in G_2 and the other random variables which are derived from the sources while a deterministic adversary A interacts with G_2. Let $I = \{i : s_i = 0\}$ and $I' = \{1,2,\ldots,q\} \setminus I$. Let $I'' = \{i \in I' : \forall j < i, P_j \neq P_i\}$. So, I'' denotes the set of all queries of A for which A_j values are sampled independently. From the definition of G_2, we have the following underlying distributions (which induces all other random variables).

Source Random Variables and Other Derived Variables

1. $Z_i \xleftarrow{\$} \{0,1\}^{nk_i}$, $\forall i \in I$;
2. $A_i \xleftarrow{\$} \{0,1\}^n$, $\forall i \in I''$ and $(C'_i, V_i, C_i) \xleftarrow{\$} \{0,1\}^{n(k_j+1)}$ $\forall i \in I'$.

Note that I, I' and I'' are also random variables which are deterministically determined from the above source variables in a recursive manner. More precisely, suppose for all $i < j$ the source random variables have been defined (i.e., conditioned). Then M_j is determined deterministically and so whether $j \in I$,

$j \in I''$ or $j \in I' \setminus I''$ is determined. Based on this, we sample the source variables on the index j. We continue it recursively for all $1 \leq j \leq q$. Now we define other dependent random variables as follows. Let $i \in I'$. We define $W_i = \mathsf{mix}_{1_R}(A_i, Q_i)$ and $D_i = \mathsf{mix}_{2_R}(V_i, W_i)$ and $Z_i = (C'_i, C_i, D_i)$. Let $Dom_j(L_i)$ denote the set $Dom(L_i)$ at the time of responding j^{th} query $i = 1, 2, 3$. So we have

1. $Dom_j(L_1) = \{P_i : i \in I', 1 \leq i < j\}$.
2. $Dom_j(L_2) = \{M_i : i \in I, i < j\} \cup \{P'_i \parallel \mathsf{mix}_{1_L}(A_i, Q_i) : i \in I', i < j\}$.
3. $Dom_j(L_3) = \{\mathsf{mix}_{2_L}(V_i, W_i) : i \in I', i < j\}$.

We denote a tuple $(x_1, \ldots, x_{j-1})$ by $x_{<j}$ or $x_{\leq j-1}$. So the j^{th} query M_j of a deterministic adversary A is a function (determined by the adversary) of the responses $Z_{<j}$ obtained so far. Now we first observe some independence among different random variables which are required to bound the probabilities of bad events. We recall that mix_1 is LL-ϵ-regular, RL-injection, LL-regular, and mix_2 is LL-ϵ-regular, LR-regular.

Lemma 5. *With the notation as defined before and the mix functions satisfying the given conditions as mentioned above we have the following:*

1. $A_{\leq j} \perp Z_{<j}$.
2. $B_i \perp Z_{\leq j}$ *for all* $i \leq j$.
3. $Z_j \perp Z_{<j}$.

Proof. 1. It is enough to show that $Z_j | A_{\leq j}, Z_{<j}$ is uniform. Moreover, we assume $j \in I'$ since otherwise it is trivially true. By definition of V_i's, the conditional distribution of V_j given $(A_{\leq j}, Z_{<j})$ is uniform. Once we fix $(A_{\leq j}, Z_{<j})$ the value of Q_j is fixed and as $\mathsf{mix}_{2_R}(\cdot, x)$ is a regular function for all x, $D_j = \mathsf{mix}_{2_R}(V_j, W_j)$ is uniform. Note that W_j depends on A_j and Q_j only. Again by definition C'_j, C_j are chosen uniformly and independently. So we are done.

2. Note that $((A_i, V_i, C'_i, C_i), (Z_k)_{k \leq j, k \neq i})$ regularly maps to $(B_i, Z_{\leq j})$ by conditioning all other source variables. This is true since A_i regularly maps to W_i (due to LR-regular property of mix_1) and (V_i, W_i) bijectively maps to B_i, D_i) (as mix_2 is a permutation). So the result follows.

3. As adversary makes distinct queries, we need to show that all outputs Z_j are independent and uniform on $\{0,1\}^{|M_j|}$. It is sufficient to show that the conditional distribution $Z_j | (Z_1, \ldots, Z_{j-1})$ is uniform on $\{0,1\}^{|M_j|}$. If $s_j = 0$ then we are done (see the line 3 of the game G_2). So assume that $s_j \neq 0$. Note that $Z_j = (C'_j, C_j, D_j)$ where $(C'_j, V_j, C_j) \xleftarrow{\$} \{0,1\}^{nk-1} \times \{0,1\}^s \times \{0,1\}^n$ (from the lines 13 and 17) and $D_j = \mathsf{mix}_{2_R}(V_j, \mathsf{mix}_{1_R}(A_j, Q_j))$. By using LR-regular property of mix_2 it is easy to see that (C'_j, C_j, D_j) is conditionally uniformly distributed. □

Types of Bad Events. Note that $\mathsf{BAD} = 1$ can occur in one of the lines 4, 10, 14, 18 and 19. Let us denote the event $\mathsf{BAD}_{j,l}$ if $\mathsf{BAD} = 1$ is set for the first time on line l of j^{th} query of the interaction A^{G_2} where $l \in \{4, 10, 14, 18, 19\}$. So

we have

$$\mathsf{Pr}[\mathsf{BAD} = 1 \text{ in the interaction } A^{G_2}] \le \sum_{j=1}^{q} \sum_{l \in \{4,10,14,18,19\}} \mathsf{Pr}[\mathsf{BAD}_{j,l}].$$

Now we bound the probabilities of the all bad events. We fix $2 \le j \le q$ as for the first query there can not be any bad event.

Claim 1. $\mathsf{Pr}[\mathsf{BAD}_{j,l}] \le (j-1)\epsilon$, $l \in \{4, 14\}$.

Proof of Claim 1. Note that $\mathsf{BAD}_{j,4}$ means that $M_j \in Dom_j(L_2)$ and so $\mathsf{mix}_{1_L}(A_i, Q_i) = X$ for some $i < j$ where the last n bits of M_j is X. Note that in lemma 5, we have shown that $A_{\le q} \perp Z_{\le q}$ and so $A_i \perp (Q_i, X)$. Thus, $\mathsf{Pr}[\mathsf{mix}_{1_L}(A_i, Q_i) = X] \le \epsilon$ (due to LL-ϵ-regular property of mix_1). Similarly, $\mathsf{BAD}_{j,14}$ means that $\mathsf{mix}_{1_L}(A_j, Q_j) = \mathsf{mix}_{1_L}(A_i, Q_i)$ for some i, and we have $(A_i, A_j) \perp (Q_i, Q_j)$ and $A_i \perp A_j$ (note that $P_i \ne P_j$). So by conditioning A_i, Q_i, Q_j and using LL-ϵ-regular property, $\mathsf{Pr}[\mathsf{mix}_{1_L}(A_j, Q_j) = \mathsf{mix}_{1_L}(A_i, Q_i)] \le \epsilon$.

Claim 2. $\mathsf{Pr}[\mathsf{BAD}_{j,10}] \le (j-1)\epsilon$.

Proof of Claim 2. Note that $\mathsf{BAD}_{j,10}$ means that $P_j \in Dom_j(L_3)$ and so there exist $i < j$ such that $B_i := \mathsf{mix}_{2_L}(V_i, W_i) = P_j$. We know that $B_i \perp Z_{\le j}$ and so $B_i \perp P_j$. By using LL-ϵ-regular property $\mathsf{Pr}[Bi = P_j] \le \epsilon$.

Claim 3. $\mathsf{Pr}[\mathsf{BAD}_{j,l}] \le (j-1)\epsilon$, $l \in \{18, 19\}$.

Proof of Claim 3. As V_j is sampled uniformly freshly, it is independent with $Z_{<j}$ and $A_{\le j}$. By using LL-ϵ-regular property, $\mathsf{Pr}[\mathsf{BAD}_{j,l}] \le (j-1)\epsilon$, $l \in \{18, 19\}$. □

By using the above bounds of bad events we have

$$\mathsf{Pr}[\mathsf{BAD} = 1 \text{ in the interaction } A^{G_2}] \le 2.5q(q-1)\epsilon.$$

5 Privacy and Integrity Security Analysis of COPA

We have already defined $\overline{\mathcal{F}}$ using $\mathcal{F}$ for full-block messages and XLS to handle the last incomplete message block. We already know that XLS is a pseudorandom function. Now we show that $\overline{\mathcal{F}}$ is a pseudorandom function and we find **modified bound** (which is also tight in terms of query complexity as described a forging algorithm in the section 3.2) for integrity security. Here we assume that $\mathcal{F}$ is an ideal random online injective function and so for all fresh queries to $\mathcal{F}$ it returns randomly keeping online property and so it returns the last block uniformly for all fresh queries. When we make a fresh $\mathcal{F}^{-1}$-query (i.e., not obtained by $\mathcal{F}$-query) then the response is $\perp$ (abort) with probability about $1 - 2^{-n}$. Moreover, $\overline{\mathcal{F}}$ uses XLS based on three independent random permutations Π_1, Π_2 and Π_3. We prove the privacy and integrity for this simplified construction. One can also extend the bound for actual construction. However, we keep this for simplified presentation of understanding of security of COPA as a domain completion.

Theorem 4. *Let $\overline{\mathcal{F}}$ be the COPA domain completion using XLS. Then we have the following results.*

1.
$$\mathbf{Adv}^{\mathrm{oprf}}_{\overline{\mathcal{F}}}(t,q,\sigma,\ell) \leq \frac{5.5q^2}{2^n}.$$

2.
$$\mathbf{Adv}^{\mathrm{auth}}_{\overline{\mathcal{F}}}(t,q,\sigma,\ell) \leq \frac{2q^3+6.5q^2+10q+1}{2^n}.$$

Proof. First part is straightforward from the composition of PRF. Note that XLS is PRF with advantage $5q^2/2^n$. Now if we have no collision on the inputs of XLS then we can not distinguish it from online random injective function. So overall online RPF advantage is bounded by $5q^2/2^n + q^2/2^{n+1}$ and hence it proves the first part.

Now we prove the second part.

Notation. Without loss of generality, we assume that adversary $\mathcal{A}$ makes queries both $M_i[1..d_i]$ and $M_i[1..d_i-1]$ where $d_i \geq 2$ and $m_i := M_i[d_i]$ is an incomplete block, $1 \leq i \leq q$. Suppose the responses are $C[1..d_i]\|T_i$ and $C_i[1..d_{i-1}]\|S_i$ respectively. We write $c_i := C_i[d_i]$. Thus, $\mathsf{XLS}(M_i[d_i]\|S_i) = T_i\|C_i[d_i] := T_i\|c_i$. Finally, it forges $C^*[1..d+1] = C^*[1..d]\|T^*$. Let us assume that $|M_i[d_i]| = n-1$ for all i. For a smaller incomplete block the similar proof works with more notational complexity. Again for the sake of notational simplicity we provide the simpler proof.

We now describe our proof strategy. Let Fail denote the event the forgery $\mathcal{A}$ fails to forge correctly. We want to show that the $\mathsf{Pr}[\mathsf{Fail}] \geq 1 - \frac{2q^3+6.5q^2+10q+1}{2^n}$. To show this, we consider the set $\mathcal{T}$ of all transcripts of the interaction $\mathcal{A}^{\overline{\mathcal{F}}}$. A transcript consists of all the values (mentioned above) observed by $\mathcal{A}$. We say that a transcript τ is **good** if the following three hold:

1. S_i's are distinct.
2. $C_i[1..d_i-1]$ are distinct.
3. For all i, j and k, we have

$$S_k \neq S_i \oplus (S_i \oplus S_j)^{\lll 2} \oplus (c_i \oplus c_j) \oplus (c_i \oplus c_j)^{\lll 2}.$$

Lemma 6. *For any good transcript τ, $\mathsf{Pr}[\mathsf{Fail} \mid \tau(\mathcal{A}^{\overline{\mathcal{F}}}) = \tau] \geq (1 - \frac{10q+1}{2^n})$.*

Proof. Proof of the lemma is postponed to the following subsection. □

Lemma 7.
$$\mathsf{Pr}[\tau(\mathcal{A}^{\overline{\mathcal{F}}}) \text{ is good }] \geq 1 - \frac{6.5q^2+2q^3}{2^n}.$$

Proof. Let O denote the ideal random online authenticated function (for variable length). Note that $\mathcal{F}$ is assumed to be same for full-block message. Now, $\mathsf{Pr}[\tau(\mathcal{A}^{\overline{\mathcal{F}}}) \text{ is good}] \geq \mathsf{Pr}[\tau(\mathcal{A}^{\mathsf{O}}) \text{ is good}] - 5.5q^2/2^n$. We use the privacy bound for the inequality. Now S_i's and c_i's are randomly distributed while $\mathcal{A}$ is interacting

with O. So by using union bound on the complement of the good transcript, we have $\mathsf{Pr}[\tau(\mathcal{A}^{\mathsf{O}}) \text{ is good}] \geq 1 - q^3/2^{n-1} - q^2/2^n$. This proves the lemma. □

From above lemmas, we have $\mathsf{Pr}[\mathsf{Fail}] \geq (1 - \frac{6.5q^2+2q^3}{2^n})(1 - \frac{10q+1}{2^n}) \geq 1 - \frac{2q^3+6.5q^2+10q+1}{2^n}$. This completes the proof of the integrity security of COPA. □

5.1 Proof of Lemma 6

We introduce some more notation. We denote A_i, U_i, V_i, and B_i to denote the all n-bit intermediate values and W_i to denote $(n-1)$-bit values for the XLS-computation of the i^{th} message (see the figure corresponding to GXLS). For the forged message we similarly denote A, U, V, B and W. We prove the lemma by considering several cases.

Case 1: $C^*[1..d-1] \neq C_i[d_i - 1]$ **for all** i: If $C^*[1..d-1] \neq C_i[d_i - 1]$ for all i then clearly forging $\mathcal{F}$ means that forging $\overline{\mathcal{F}}$. So $\mathsf{Pr}[\mathcal{A} \text{ forges }] \leq 2^{-n}$.

Case 2: $C^*[1..d-1] = C_i[d_i - 1]$**,** $T^* = T_j$ **for some** i **and** j: The condition 3 for a good transcript forces that $V_k \neq V$ for all k. Hence, after conditioning on all inputs-outputs of Π_1, Π_2, Π_3 favorable to the transcript, the output of $\Pi_2^{-1}(V) = U$ is uniformly distributed over a set of size at lease $2^n - q$. Since U to A mapping is defined by $\mathsf{mix2}_L$ which is a 2-to-one mapping. So, at most $2q$ choices of A values can collide with A_k for some k. Thus, with probability at least $1 - 3q/2^n$, A is different from all A_k values. So $m^*\|S^*$ is uniformly distributed over a set of size at least $(2^n - q)(2^{n-1} - 3q)$. This proves that $\mathsf{Pr}[S^* \neq S_i \mid A \text{ is fresh }] \geq 1 - 4q/2^n$ and so the unconditional probability $\mathsf{Pr}[S^* \neq S_i] \geq 1 - 7q/2^n$. Hence

$$\mathsf{Pr}[\mathsf{Fail} \mid \tau(\mathcal{A}^{\overline{\mathcal{F}}}) = \tau] \geq (1 - (7q+1)/2^n).$$

Case 3: $C^*[1..d-1] = C_i[d_i - 1]$**, for some** i**,** $T^* \neq T_j$ **for all** j: This case is also similar to the previous case. However, we need to start the argument of freshness in the inputs from Π_1^{-1} onwards. As T^* is fresh to Π_3^{-1}, the output B is uniformly distributed over a set of size at least $2^n - q$. Among all these choices, at least $2^n - 3q$ would give a fresh V input to Π_2^{-1}. Similarly, at least $2^n - 3q$ choices of A are fresh given that V is fresh. Finally, $m^*\|S^*$ is uniformly distributed over a set of size at least $(2^n - q)(2^{n-1} - 3q)$. By combining all these probabilities we have

$$\mathsf{Pr}[\mathsf{Fail} \mid \tau(\mathcal{A}^{\overline{\mathcal{F}}}) = \tau] \geq (1 - (10q+1)/2^n).$$

6 Conclusion and Future Works

In this paper we first show a distinguishing attack on XLS. We extend this attack to obtain a forging algorithm on COPA. Our forging algorithm makes about $2^{n/3}$ queries. This algorithm can be applied to all similar authenticated encryptions which handle the partial block in a similar fashion. In a positive side,

we show that XLS is a pseudorandom function. In fact, XLS is little more stronger than pseudorandom function. By exploiting this, the authenticated encryption COPA can be shown to be secure up to $2^{n/3}$ queries. Thus, the query complexity bound for COPA is tight. However, in terms of time complexity, it requires to solve generalized birthday problem in which case we do not know any efficient algorithm for three lists. Thus, it remains open problem to analyze COPA in terms of time complexity.

References

1. CAESAR submissions, 2014. http://competitions.cr.yp.to/caesar-submissions.html.
2. ISO/IEC 9797. Data cryptographic techniques-Data integrity mechanism using a cryptographic check function employing a blockcipher algorithm, 1989.
3. Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. Parallelizable and authenticated online ciphers. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 424–443. Springer, 2013.
4. M. Bellare, A. Boldyreva, L. Knudsen, and C. Namprempre. On-line ciphers and the hash-cbc constructions. In *Advances in Cryptology –Crypto 2001*, number 2139 in Lecture Notes in Computer Science, pages 292–309, Berlin, 2001. Springer.
5. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The Security of the Cipher Block Chaining Message Authentication Code. *J. Comput. Syst. Sci.*, 61(3):362–399, 2000.
6. Anne Canteaut, editor. *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*. Springer, 2012.
7. Larry Carter and Mark N. Wegman. Universal Classes of Hash Functions. *J. Comput. Syst. Sci.*, 18(2):143–154, 1979.
8. Debrup Chakraborty and Palash Sarkar. HCH: A new tweakable enciphering scheme using the hash-encrypt-hash approach. In Rana Barua and Tanja Lange, editors, *INDOCRYPT*, volume 4329 of *Lecture Notes in Computer Science*, pages 287–302. Springer, 2006.
9. Debra L. Cook, Moti Yung, and Angelos D. Keromytis. Elastic aes. *IACR Cryptology ePrint Archive*, 2004:141, 2004.
10. Debra L. Cook, Moti Yung, and Angelos D. Keromytis. Elastic block ciphers: method, security and instantiations. *Int. J. Inf. Sec.*, 8(3):211–231, 2009.
11. Joan Daemen and Vincent Rijmen. The Design of Rijndael: AES - The Advanced Encryption Standard., 2002. http://csrc.nist.gov/CryptoToolkit/aes/rijndael/Rijndael-ammended.pdf.
12. Ivan Damgard. A design principle for hash functions. In Gilles Brassard, editor, *Advances in Cryptology CRYPTO 89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer New York, 1990.
13. Morris Dworkin. Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. http://csrc.nist.gov/publications/nistpubs/index.html#sp800-38B.

14. Ewan Fleischmann, Christian Forler, and Stefan Lucks. Mcoe: A family of almost foolproof on-line authenticated encryption schemes. In Canteaut [6], pages 196–215.
15. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, August 1986.
16. Shai Halevi. EME*: Extending EME to handle arbitrary-length messages with associated data. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT*, volume 3348 of *Lecture Notes in Computer Science*, pages 315–327. Springer, 2004.
17. Shai Halevi. TET: A wide-block tweakable mode based on Naor-Reingold. Cryptology ePrint Archive, Report 2007/014, 2007. http://eprint.iacr.org/.
18. Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 482–499. Springer, 2003.
19. Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer, 2004.
20. Tadayoshi Kohno, John Viega, and Doug Whiting. Cwc: A high-performance conventional authenticated encryption mode. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 408–426. Springer, 2004.
21. M. Luby and C. Rackoff. How to construct pseudo-random permutations from pseudo-random functions. In *Advances in Cryptology – Crypto 1985*, number 218 in Lecture Notes in Computer Science, page 447, New York, 1984. Springer-Verlag.
22. David A. McGrew and Scott R. Fluhrer. The extended codebook (XCB) mode of operation. Cryptology ePrint Archive, Report 2004/278, 2004. http://eprint.iacr.org/.
23. RalphC. Merkle. A certified digital signature. In Gilles Brassard, editor, *Advances in Cryptology CRYPTO 89 Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer New York, 1990.
24. Mridul Nandi. Two new efficient cca-secure online ciphers: Mhcbc and mcbc. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *INDOCRYPT*, volume 5365 of *Lecture Notes in Computer Science*, pages 350–362. Springer, 2008.
25. Mridul Nandi. A generic method to extend message space of a strong pseudorandom permutation. *Computación y Sistemas*, 12(3), 2009.
26. Mridul Nandi. XLS is not a strong pseudorandom permutation. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 478–490. Springer, 2014.
27. Thomas Ristenpart and Phillip Rogaway. How to enrich the message space of a cipher. In Alex Biryukov, editor, *FSE*, volume 4593 of *Lecture Notes in Computer Science*, pages 101–118. Springer, 2007.
28. Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *ACM Conference on Computer and Communications Security*, pages 98–107. ACM, 2002.
29. Phillip Rogaway and Thomas Shrimpton. The siv mode of operation for deterministic authenticated-encryption (key wrap) and misuse-resistant nonce-based authenticated-encryption. *Aug*, 20:3, 2007.

30. Phillip Rogaway, Mark Wooding, and Haibin Zhang. The security of ciphertext stealing. In Canteaut [6], pages 180–195.
31. Phillip Rogaway and Haibin Zhang. Online ciphers from tweakable blockciphers. In Aggelos Kiayias, editor, *CT-RSA*, volume 6558 of *Lecture Notes in Computer Science*, pages 237–249. Springer, 2011.
32. Palash Sarkar. Improving upon the tet mode of operation. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *ICISC*, volume 4817 of *Lecture Notes in Computer Science*, pages 180–192. Springer, 2007.
33. Peng Wang, Dengguo Feng, and Wenling Wu. HCTR: A variable-input-length enciphering mode. In Dengguo Feng, Dongdai Lin, and Moti Yung, editors, *CISC*, volume 3822 of *Lecture Notes in Computer Science*, pages 175–188. Springer, 2005.
34. Mark N. Wegman and Larry Carter. New hash functions and their use in authentication and set equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981.
35. Moti Yung, editor. *A Generalized Birthday Problem*, volume 2442 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2002.

Appendix A: Where the proof of [27] went wrong

In this section we first see where the proof of XLS went wrong. Like many other proofs, authors defined several games. Game G_0 exactly simulates XLS (based on random permutations E and $\mathcal{E}$) and its inverse while G_1 always returns random bits. While interacting with these games, an event *bad* set true depending on some internal collisions. As G_0 and G_1 are identical until *bad*, for any adversary $\mathcal{A}$ we have

$$\mathbf{Adv}^{\pm\mathrm{prf}}_{\mathsf{XLS}}(\mathcal{A}) \leq \mathsf{Pr}[\mathcal{A}^{G_1} \text{ sets bad}].$$

The games $G_2.G_3$ and G_4 are defined and it had been shown that

$$\mathsf{Pr}[\mathcal{A}^{G_1} \text{ sets bad}] \leq \mathsf{Pr}[\mathcal{A}^{G_4} \text{ sets bad}] + p$$

for some negligible p. In the game G_4 the adversary specifies a transcript $\tau = (\mathbf{M}, \mathbf{C}, \mathbf{ty})$ where $\mathbf{M} = (\mathtt{M}_1, \mathtt{M}_2, \ldots, \mathtt{M}_\mathtt{q})$, $\mathbf{C} = (\mathtt{C}_1, \mathtt{C}_2, \ldots, \mathtt{C}_\mathtt{q})$ and $\mathbf{ty} = (\mathtt{ty}_1, \mathtt{ty}_2, \ldots, \mathtt{ty}_\mathtt{q})$ where $\mathtt{ty}_\mathtt{i}$'s are either Enc or Dec. If it is Enc then the i^{th} query is $\mathtt{M}_\mathtt{i}$ and $\mathtt{C}_\mathtt{i}$ is the response. Otherwise, the i^{th} query is $\mathtt{C}_\mathtt{i}$ and $\mathtt{M}_\mathtt{i}$ is the response. The internal variables are defined in such a manner so that the above transcript is realized. *The bad events holds true if a collision on inputs or outputs of $\mathcal{E}$ occurs.* In Claim 7 and 8 of [27], it is shown that $\mathsf{Pr}[\mathcal{A}^{G_4} \text{ sets bad}] \leq q^2/2^{n+1}$ when we set $s = n - 1$. However, the claims are incorrect. We describe a transcript of four queries as follows:

1. $\mathbf{M} = ((P, Q), (P, Q \oplus \alpha), (P_3, Q_3), (P_4, Q_3 \oplus \gamma)$,
2. $\mathbf{C} = ((C, D), (C', D \oplus \beta), (C, D_3), (C', D_3 \oplus \gamma)$ and
3. $\mathbf{ty} = (\mathsf{Enc}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Dec})$

where $\gamma = (\alpha \oplus \beta) \oplus (\alpha \oplus \gamma)^{\ggg 2}$. This is exactly same as one of the possible transcript of our distinguisher $\mathcal{A}_0$. As discussed in the analysis of $\mathcal{A}_0$, the collisions on outputs of $\mathcal{E}$ (in the last two queries) occur with probability 1/2. So the claims are not correct.

Now we investigate where exactly it went wrong. In the proof of claim 7, authors considered six cases. In the third case there is one sub-case which considers $M_2^i \neq M_2^j$ (the inputs of first E call of an encryption query). Subsequently, authors mistakenly claimed that

"$M_4^i, M_5^i, M_4^j, M_5^j$ *are all independently and uniformly chosen at random*".

Note that for a specific transcript, M_2^i and M_2^j can appear as parts of the outputs of two previous decryption queries. In this case, M_5^i and M_5^j are defined through the mix function instead of sampling. So these can be very much dependent for some choices of transcripts.

Appendix B: Figures

We illustrate XLS, GXLS and 3-round of elastic blockcipher in Fig. 6.

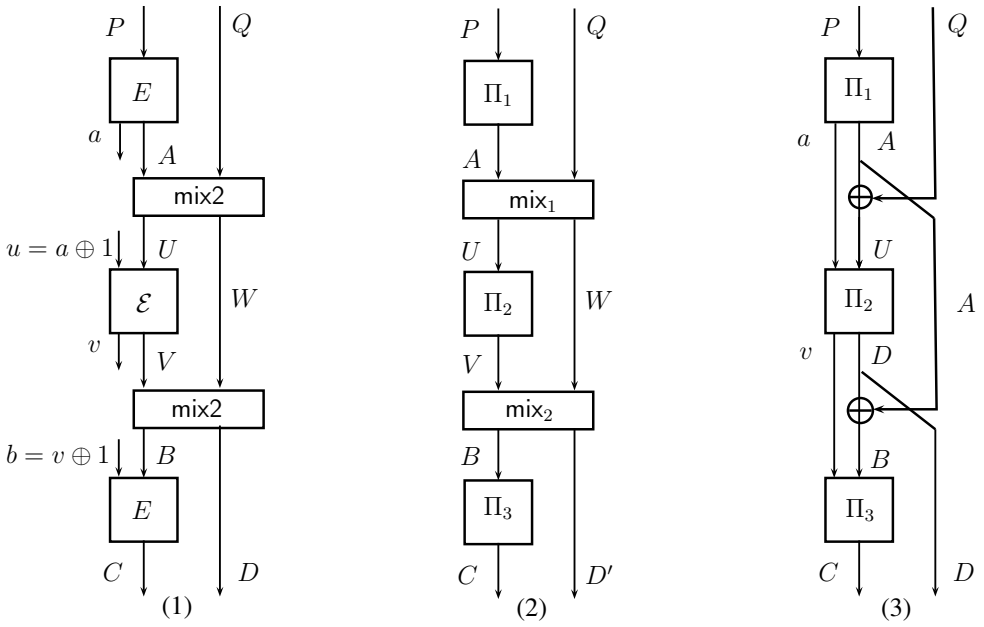

Fig. 4. Illustration of (1) XLS, (2) GXLS and (3) 3-round Elastic Blockcipher. The XLS and 3-round Elastic blockcipher have specific mix functions mix_1 and mix_2 for all $1 \leq s \leq n-1$.

Appendix C: Description of the Games

Game G_2 $\boxed{G_1}$ (with boxed statements executed).

Initialization: partial functions $L_1 = L_2 = L_3 = \phi$ (empty), $j = 0$, $\mathsf{BAD} = 0$.

On query $M := M_{j+1}$

1 $j = j + 1$ and let $|M_j| = nk_j + s_j$, $0 \leq s_j < n$, $k \geq 1$
2 **if** $s_j = 0$ **then**
3 $\quad Z_j \xleftarrow{\$} \{0,1\}^{|M_j|}$.
4 $\quad$ **if** $M_j \in Dom(L_2)$ **then** $\mathsf{BAD}[2,2] = 1$, $\boxed{Z_j = L_2(M_j)}$.
5 $\quad$ **return** $L_2(M_j) \leftarrow Z_j$.
6 **else**
7 $\quad$ **parse** $M_j = (P'_j, P_j, Q_j) \in \{0,1\}^{n(k-1)} \times \{0,1\}^n \times \{0,1\}^s$
8 $\quad A_j \xleftarrow{\$} \{0,1\}^n$.
9 $\quad$ **if** $P_j \in Dom(L_1)$ **then** $A_j \leftarrow L_1(P_j)$.
10 $\quad$ **else if** $P_j \in Dom(L_3)$ **then** $\mathsf{BAD}[1,3] = 1$, $\boxed{A_j \leftarrow L_3(P_j)}$
11 $\quad$ **else** $L_1(P_j) \leftarrow A_j$.
12 $\quad (U_j, W_j) \leftarrow \mathsf{mix}_1(A_j, Q_j)$.
13 $\quad (C'_j, V_j) \xleftarrow{\$} \{0,1\}^{n(k-1)} \times \{0,1\}^n$
14 $\quad$ **if** $(P'_j, U_j) \in Dom(L_2)$ **then**
$\quad\quad \mathsf{BAD}[2,2] = 1$, $\boxed{(C'_j, V_j) = L_2(P'_j, U_j)}$.
15 $\quad$ **else** $L_2(P'_j, U_j) \leftarrow (C'_j, V_j)$
16 $\quad (B_j, D_j) \leftarrow \mathsf{mix}_2(V_j, W_j)$.
17 $\quad C_j \xleftarrow{\$} \{0,1\}^n$.
18 $\quad$ **if** $B_j \in Dom(L_1)$ **then** $\mathsf{BAD}[3,1] = 1$, $\boxed{C_j \leftarrow L_1(B_j)}$.
19 $\quad$ **else if** $B_j \in Dom(L_3)$ **then** $\mathsf{BAD}[3,3] = 1$, $\boxed{C_j \leftarrow L_3(B_j)}$.
20 $\quad$ **else** $L_3(B_j) \leftarrow C_j$.
21 $\quad$ **return** $Z_j = (C'_j, C_j, D_j)$.

Algorithm 3: The game G_1 is equivalent to $G'_1 = \mathsf{GXLS}^{\mathtt{R}_1,\mathtt{R}_2}$ as $L_1 \cup L_3$ and G_2 behave like independent random functions (see lemma 2).