Another Look at XCB

Debrup Chakraborty[1], Vicente Hernandez-Jimenez[1], Palash Sarkar[2]

[1] Department of Computer Science, CINVESTAV-IPN,
Av. IPN 2508 San Pedro Zacatenco, Mexico City 07360, Mexico
debrup@cs.cinvestav.mx, vicente.845@gmail.com
[2] Applied Statistics Unit
Indian Statistical Institute
203 B.T. Road, Kolkata 700108, India
palash@isical.ac.in

Abstract. XCB is a tweakable enciphering scheme (TES) which was first proposed in 2004. The scheme was modified in 2007. We call these two versions of XCB as XCBv1 and XCBv2 respectively. XCBv2 was later proposed as a standard for encryption of sector oriented storage media in IEEE-std 1619.2 2010. There is no known proof of security for XCBv1 but the authors provided a concrete security bound for XCBv2 and a "proof" for justifying the bound. In this paper we show that XCBv2 is not secure as a TES by showing an easy distinguishing attack on it. For XCBv2 to be secure, the message space should contain only messages whose lengths are multiples of the block length of the block cipher. For such restricted message spaces also, the bound that the authors claim is not justified. We show this by pointing out some errors in the proof. For XCBv2 on full block messages, we provide a new security analysis. The resulting bound that can be proved is much worse than what has been claimed by the authors. Further, we provide the first concrete security bound for XCBv1, which holds for all message lengths. In terms of known security bounds, both XCBv1 and XCBv2 are worse compared to existing alternative TES.

1 Introduction

Tweakable enciphering schemes (TES) are a class of block-cipher mode of operation, which are meant to provide security as that of a tweakable strong pseudorandom permutation (SPRP). Designing efficient TES which provides the required security in provable terms is a challenging problem. In the last decade there have been some intense activities in designing such schemes and proving their security [3–8,11,13–16]. TES are also practically interesting, as they are the most suitable cryptographic schemes for the application of disk encryption (or encryption of any storage media which are organized as sectors). This practical side of TES has lead to standardization activities for the application of disk encryption [1]. In this paper we take a close look at a TES called XCB which is a part of the IEEE Std 1619.2-2010 standard for disk encryption.

McGrew and Fluhrer proposed XCB in [11]. However, they did not give a proof of security for their construction. Later in [12] they made changes to the original construction proposed in [11] and proved security of the updated construction. In [12] the authors claim that the changes made to the original construction help in an easy analysis of it. For ease of reference we shall call the different versions of XCB in [11] and [12] as XCBv1 and XCBv2 respectively.

In this paper we do a careful analysis of both XCBv1 and XCBv2. As a result of our analysis we conclude that the security claims about XCBv2 made in [12] are largely erroneous. Primarily, XCBv2 is not at all secure for messages whose lengths are not the multiples of the block length of the underlying block cipher. We demonstrate this by a simple distinguishing attack on XCBv2. If we restrict the message space to contain only messages whose lengths are multiples of the block length of the block cipher then our distinguishing attack does not work. But in case of such restricted message spaces also, the security bound claimed by the authors is not justified. We demonstrate this by pointing out an error in the security proof of XCBv2. The error occurs in a result (Theorem

1 in [12]) which is central to the proof of the security theorem stated in [12]. We were not able to construct a counterexample for the security theorem itself, but surely the proof provided for the security theorem is incorrect.

When the message space is restricted to full block messages, we also provide a new security theorem for XCBv2, where the security bound that can be proved is significantly greater than that claimed in [12]. Additionally, we prove security for the original XCB (i.e. XCBv1) as proposed in [11]. Our security theorem for XCBv1 provides a concrete security bound which was not known before. The proof of XCBv1 and the bound is almost same as that of XCBv2.

XCBv2 with AES as the underlying block cipher has been accepted by the IEEE as a standard (IEEE-std 1619.2, 2010) for wide block encryption for shared storage media [1]. Being a part of the standard it is expected that XCBv2 would be soon deployed (or has been already deployed) as an encryption scheme for a wide range of storage applications. Our results show that XCBv2 has numerous security flaws, thus the users and implementors should be cautions regarding its use. Moreover our analysis puts in serious doubt the process and outcome of the IEEE working group on security in storage, the group which proposed this standard.

Notation. In what follows, we shall denote the set of all n bit strings by $\{0,1\}^n$. For binary strings X and Y, $X||Y$ will denote the concatenation of X and Y; $|X|$ will denote the length of X in bits; $\mathsf{msb}_r(X)$ and $\mathsf{lsb}_r(X)$ will denote the r most significant and least significant bits of X respectively. By $\mathsf{int}(X)$ we would mean the integer represented by the binary string X, and for a non-negative integer $i \leq 2^n - 1$, $\mathsf{bin}_n(i)$ will denote the n-bit binary representation of i. We shall often treat n-bit strings as elements in $GF(2^n)$, thus, for $X, Y \in \{0,1\}^n$, $X \oplus Y$ and XY will respectively denote addition and multiplication in $GF(2^n)$.

2 Differences between XCBv2 and XCBv1

The encryption algorithms of XCBv1 and XCBv2 are described in Fig. 1. In Fig. 1, for a binary string X, $\mathsf{parse}_n(X)$ outputs $X_1, X_2, \ldots, X_m$, where $X = X_1||X_2||\ldots||X_m$ and that $|X_i| = n$ bits $(i = 1, 2, \ldots m-1)$, $0 < |X_m| \leq n$.

The hash function H in Fig. 1 is defined as

$$\begin{aligned} H_h(X,T) =& X_1 h^{m+p+1} \oplus X_2 h^{m+p} \oplus \ldots \oplus \mathsf{pad}(X_m) h^{p+2} \oplus T_1 h^{p+1} \\ & \oplus T_2 h^p \oplus \ldots \oplus \mathsf{pad}\,(T_p)\, h^2 \oplus (\mathsf{bin}_{\frac{n}{2}}(|X|)||\mathsf{bin}_{\frac{n}{2}}(|T|))h, \end{aligned} \tag{1}$$

where h is an n-bit hash key and $(X_1, X_2, \ldots, X_m) = \mathsf{parse}_n(X)$, $(T_1, T_2, \ldots, T_p) = \mathsf{parse}_n(T)$. The pad function is defined as $\mathsf{pad}(X_m) = X_m||0^r$ where $r = n - |X_m|$. Thus, $|\mathsf{pad}(X_m)| = n$.

Given an n-bit string S, the counter mode Ctr is defined as

$$\mathsf{Ctr}_{K,S}(A_1, \ldots, A_m) = (A_1 \oplus E_K(\mathsf{inc}^0(S)), \ldots, A_m \oplus E_K(\mathsf{inc}^{m-1}(S))).$$

In case the last block A_m is incomplete then $A_m \oplus E_K(\mathsf{inc}^{m-1}(S))$ in Ctr is replaced by $A_m \oplus \mathsf{drop}_r(E_K(\mathsf{inc}^{m-1}(S)))$, where $r = n - |A_m|$ and $\mathsf{drop}_r(E_K(\mathsf{inc}^{m-1}(S)))$ is the first $(n-r)$ bits of $E_K(\mathsf{inc}^{m-1}(S))$. In the definition of Ctr, for a bit string $X \in \{0,1\}^n$, $\mathsf{inc}(X)$ treats the least significant 32 bits (the rightmost 32 bits) of X as a non-negative integer, and increments this value modulo 2^{32}, i.e.,

$$\mathsf{inc}(X) = \mathsf{msb}_{n-32}(X)||\mathsf{bin}_{32}(\mathsf{int}(\mathsf{lsb}_{32}(X)) + 1 \bmod 2^{32}).$$

For $r \geq 0$, we write $\mathsf{inc}^r(X)$ to denote the r times iterative applications of inc on X. We use the convention that $\mathsf{inc}^0(X) = X$.

For both XCBv1 and XCBv2, it is specified (in [11] and [12] respectively) that $n \leq |P| \leq 2^{39}$ and $0 \leq |T| \leq 2^{39}$. The description of XCB in the standard 1619.2 is same as the description of XCBv2 (but it seems that there is a typo in the description, we discuss more about it in Section 2.1). In the standard the block cipher is fixed as AES (with either 128-bit key ot 256-bit key), hence $n = 128$. And it it is specified that tweaks can be of arbitrary length and message lengths should always be multiples of 8 bits and can be between 128 bits and 2^{32} bits.

Next we point out some of the main differences in the two versions of XCB.

1. **Only a single hash key is used**. XCBv1 has a different key for each computation of the hash function. In XCBv2 the computation of the hash function is done using the same key. This change was justified in [12] by saying that a single hash enables the derivation of some algebraic relations between the two hash functions, which in turn helps in the security proof. Moreover, a single hash key benefits software implementations by relieving them of the need to store pre-computed tables for an additional hash key[3].
2. **Key sizes**. Both XCBv1 and XCBv2 are parameterized by the key K which is the key of the underlying block cipher. For using XCBv1 it has to be the case that the block length of the block cipher and the key lengths are the same. Notice, that in XCBv1 both K_d and K_c are block cipher outputs and are thus bound to be of the same length of that of the block length of the block cipher. Thus in XCBv1 one cannot use an AES with 192 or 256-bit keys. This limitation was first pointed out in [4]. In XCBv2 this restriction has been removed and it can work for any block cipher whose key length is less than or equal to twice its block length.
3. **Inputs to the hash function are rearranged**. The order of the inputs to the hash in the two different constructions differ along with some extra formatting of the inputs. In XCBv1 first and second computations of the hash function in lines 7 and 9 are the same, except that in line 7 the input to the hash is the tweak and the plaintext and in line 9 the inputs are the tweak and the ciphertext. In XCBv2 the two computations of the hash functions (in lines 108 and 110) differ in the following ways:
   (a) In line 108, the first input to the hash is the tweak with a block of zeros concatenated in the beginning, whereas in line 110 the first input is the tweak with a block of zeros concatenated in the end.
   (b) The second input in line 108 is the padded plain text concatenated with a block of zeros. And in line 110, the second input is the padded ciphertext concatenated with the length of the tweak plus the block length, and the length of the ciphertext.

   Note that in case of XCBv2 the second inputs to both the hashes are already padded, and the length parameter is only added in the second hash (line 110).

   According to the authors this change in the formatting and order of the hash inputs helps in their analysis, but given that the original definition of the hash (in Eq. (1)) already includes a length parameter of the inputs, it is not clear why the authors propose to input the padded messages thus nullifying the effect of the lengths in the first hash and again adding the extra length parameter in case of the second hash computation. We will see later this makes XCBv2 insecure for certain types of messages.
4. **Plaintext is formatted differently**. XCBv1 uses the first n bits of the plaintext to compute the value CC while XCBv2 uses the last n bits of the plaintext to compute the value of the same variable. According to the authors, this change makes the construction amenable to a pipelined implementation.

2.1 Typo in the Standard?

The specification in the standard IEEE 1619.2 considers only 128-bit blocks and specifies the underlying block cipher as AES with key size of either 128 bits or 256 bits. The specification of XCB as presented in the standard is shown in Section A of the appendix. Our description of XCBv2 in Fig. 1 not only follows a different notation but it is otherwise different from the description in the standard.

The encryption scheme specified in the standard document is *not length preserving* and it would lead to a 2-bit expansion in the ciphertext, i.e., the ciphertext produced by the encryption scheme would be two bits longer than the plaintext. Also, the decryption algorithm is not the inverse of the encryption algorithm. Which means that XCB as specified in the standard is not even an encryption scheme.

We believe that these problems occurs due to a typo. Line 6 should be $B \leftarrow P[0 : m - 129]$ instead of $B \leftarrow P[0 : m - 127]$. With this change, the scheme becomes a special case (with $n = 128$ and the aforementioned message length restrictions) of the description of XCB in [12] (which is same as the description of XCBv2 in Fig. 1).

[3] While arguing about efficiency of XCB the authors stress on a software implementation of the multiplier which uses pre-computed tables, and in such a software implementation only XCB *may have* its efficiency comparable with constructions which only uses block ciphers. It is known that in hardware XCB performs worse than all known efficient TES [10].

Encryption under XCBv1: $\mathbf{E}_K^T(P)$

0. $(P_1, \ldots, P_m) \leftarrow \mathsf{parse}_n(P)$
1. $h_1 \leftarrow E_K(0^{n-3}||001)$
2. $h_2 \leftarrow E_K(0^{n-3}||011)$
3. $K_e \leftarrow E_K(0^n)$
4. $K_d \leftarrow E_K(0^{n-3}||100)$
5. $K_c \leftarrow E_K(0^{n-3}||010)$
6. $CC \leftarrow E_{K_e}(P_1)$
7. $S \leftarrow CC \oplus H_{h_1}(P_2|| \ldots ||P_{m-1}||P_m, T)$
8. $(C_2, \ldots, C_m) \leftarrow \mathsf{Ctr}_{K_c,S}(P_2, \ldots, P_m)$
9. $MM \leftarrow S \oplus H_{h_2}(C_2|| \ldots ||C_{m-1}||C_m, T)$
10. $C_1 \leftarrow E_{K_d}^{-1}(MM)$
11. return $(C_1, C_2, \ldots, C_m)$

Encryption under XCBv2: $\mathbf{E}_K^T(P)$

100. $P_m \leftarrow \mathsf{lsb}_n(P)$
101. $A \leftarrow \mathsf{msb}_{|P|-n}(P)$
102. $(P_1, P_2, \ldots, P_{m-2}, P_{m-1}) \leftarrow \mathsf{parse}_n(A)$
103. $h \leftarrow E_K(0^n)$
104. $K_e \leftarrow \mathsf{msb}_{|K|}(E_K(0^{n-3}||001)||E_K(0^{n-3}||010))$
105. $K_d \leftarrow \mathsf{msb}_{|K|}(E_K(0^{n-3}||011)||E_K(0^{n-3}||100))$
106. $K_c \leftarrow \mathsf{msb}_{|K|}(E_K(0^{n-3}||101)||E_K(0^{n-3}||110))$
107. $CC \leftarrow E_{K_e}(P_m)$
108. $S \leftarrow CC \oplus H_h(0^n||T, P_1|| \ldots ||P_{m-2}||\mathsf{pad}(P_{m-1})||0^n)$
109. $(C_1, \ldots, C_{m-1}) \leftarrow \mathsf{Ctr}_{K_c,S}(P_1, \ldots, P_{m-2}, P_{m-1})$
110. $MM \leftarrow S \oplus H_h(T||0^n, C_1|| \ldots ||\mathsf{pad}(C_{m-1})||(\mathsf{bin}_{\frac{n}{2}}(|T||0^n|)||\mathsf{bin}_{\frac{n}{2}}(|C_1|| \ldots ||C_{m-2}||C_{m-1}|)))$
111. $C_m \leftarrow E_{K_d}^{-1}(MM)$
112. return $(C_1, C_2, \ldots, C_{m-1}, C_m)$

Fig. 1. Encryption using XCBv1 and XCBv2.

3 Security of Tweakable Enciphering Schemes

The discussion in this section is based on [8]. An n-bit block cipher is a function $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, where $\mathcal{K} \neq \emptyset$ is the key space and for any $K \in \mathcal{K}$, $E(K,.)$ is a permutation. We write $E_K(\,)$ instead of $E(K,.)$.

An adversary A is a probabilistic algorithm which has access to some oracles and which outputs either 0 or 1. Oracles are written as superscripts. The notation $A^{\mathcal{O}_1,\mathcal{O}_2} \Rightarrow 1$ denotes the event that the adversary A, interacts with the oracles $\mathcal{O}_1, \mathcal{O}_2$, and finally outputs the bit 1. In what follows, by the notation $X \xleftarrow{\$} \mathcal{S}$, we will denote the event of choosing X uniformly at random from the finite set $\mathcal{S}$.

Let $\mathrm{Perm}(n)$ denote the set of all permutations on $\{0,1\}^n$. We define two security notions of $E(.,.)$. The advantage of an adversary A in breaking the pseudorandomness of $E(.,.)$ is defined as

$$\mathbf{Adv}_E^{\mathrm{prp}}(A) = \left|\Pr\left[K \xleftarrow{\$} \mathcal{K} : A^{E_K(\,)} \Rightarrow 1\right] - \Pr\left[\pi \xleftarrow{\$} \mathrm{Perm}(n) : A^{\pi(\,)} \Rightarrow 1\right]\right|,$$

and the advantage of A in breaking the strong pseudorandomness of E is defined as

$$\mathbf{Adv}_E^{\pm\mathrm{prp}}(A) = \Big|\Pr\left[K \xleftarrow{\$} \mathcal{K} : A^{E_K(\,),E_K^{-1}(\,)} \Rightarrow 1\right] - \Pr\left[\pi \xleftarrow{\$} \mathrm{Perm}(n) : A^{\pi(\,),\pi^{-1}(\,)} \Rightarrow 1\right]\Big|.$$

A tweakable enciphering scheme is a function $\mathbf{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$, where $\mathcal{K} \neq \emptyset$ and $\mathcal{T} \neq \emptyset$ are the key space and the tweak space respectively. The message and the cipher spaces are $\mathcal{M}$. We shall write $\mathbf{E}_K^T(.)$ instead of $\mathbf{E}(K,T,.)$. The inverse of an enciphering scheme is $\mathbf{D} = \mathbf{E}^{-1}$ where $X = \mathbf{D}_K^T(Y)$ if and only if $\mathbf{E}_K^T(X) = Y$.

Let $\mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ denote the set of all functions $\boldsymbol{\pi} : \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ where $\boldsymbol{\pi}(\mathcal{T},.)$ is a length preserving permutation. Such a $\boldsymbol{\pi} \in \mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ is called a tweak indexed permutation. For a tweakable enciphering scheme $\mathbf{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$, we define the advantage an adversary A has in distinguishing $\mathbf{E}$ and its inverse from a random tweak indexed permutation and its inverse in the following manner.

$$\mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(A) = \Big|\Pr\left[K \xleftarrow{\$} \mathcal{K} : A^{\mathbf{E}_K(.,.),\mathbf{E}_K^{-1}(.,.)} \Rightarrow 1\right] - \Pr\left[\boldsymbol{\pi} \xleftarrow{\$} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M}) : A^{\boldsymbol{\pi}(.,.),\boldsymbol{\pi}^{-1}(.,.)} \Rightarrow 1\right]\Big|. \tag{2}$$

Here, $\boldsymbol{\pi} \xleftarrow{\$} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ means that for each ℓ such that $\{0,1\}^\ell \subseteq \mathcal{M}$ and $T \in \mathcal{T}$ we choose a tweakable random permutation π^T from $\mathrm{Perm}(\ell)$ independently. We define $\mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(q,\sigma)$ by $\max_A \mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(A)$ where maximum is taken over all adversaries which makes at most q queries having at most σ many blocks. For a computational advantage we define $\mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(q,\sigma,t)$ by $\max_A \mathbf{Adv}_{\mathbf{E}}^{\pm\widetilde{\mathrm{prp}}}(A)$. In addition to the previous restrictions on A, it can run in time at most t.

Pointless queries: Let T, P and C represent tweak, plaintext and ciphertext respectively. We assume that an adversary never repeats a query, i.e., it does not ask the encryption oracle with a particular value of (T,P) more than once and neither does it ask the decryption oracle with a particular value of (T,C) more than once. Furthermore, an adversary never queries its deciphering oracle with (T,C) if it got C in response to an encipher query (T,P) for some P. Similarly, the adversary never queries its enciphering oracle with (T,P) if it got P as a response to a decipher query of (T,C) for some C. These queries are called *pointless* as the adversary knows what it would get as responses for such queries.

The notation $\mathbf{E}[E]$ denotes a tweakable enciphering scheme $\mathbf{E}$, where the underlying n-bit block cipher is E. Both XCBv2 and XCBv1 utilizes three block ciphers with different keys. In the ideal scenario, these three block ciphers can be considered as three independent random permutations. We shall denote XCBv2 instantiated with three permutations drawn uniformly at random from $\mathrm{Perm}(n)$ and a hash key h drawn uniformly at random from $\{0,1\}^n$ as XCBv2[3Perm(n)]. We will do the same for XCBv1, i.e., XCBv1[3Perm(n)] will denote XCBv1 instantiated with three random permutations and the two hash keys h_1, h_2 drawn uniformly from $\{0,1\}^n$.

4 Security Claims in [12]

The security of XCB as claimed in [12] is stated below using our (and standard) notations and terminology[4].

Theorem 1. *Let A be an arbitrary adversary attacking XCBv2[3Perm(n)] who asks at most q queries each of length at most l bits (where $l \geq 2n$). Then*

$$\mathbf{Adv}^{\pm\widetilde{\text{prp}}}_{\text{XCBv2[3Perm}(n)]}(A) \leq \frac{q^2\lceil l/n+2\rceil^2 2^3}{2^n}.$$

The bound stated in the above theorem is based on the following result:

Theorem 2 (H is unlikely to collide with $\mathsf{inc}^s(H)$). *For any T, T', P, P', D, D' where either $T' \neq T$ or $P' \neq P$ or both inequalities hold, and any index s,*

$$\Pr[h \xleftarrow{\$} \mathcal{K} : H_h(T,P) \oplus D = \mathsf{inc}^s(H_h(T',P') \oplus D')] \leq \frac{\lceil l/n+2\rceil}{2^n}$$

whenever the inputs T and P are restricted to so that the sum of their lengths is l or fewer bits.

In the following two sections we show that both these Theorems are wrong by showing counter examples.

5 Distinguishing attack on XCBv2

Here we give a distinguishing attack on XCBv2 which violates Theorem 1.

A distinguishing attack on a TES consists of the construction of an adversary which can distinguishing between the scheme and a random permutation with high probability. The existence of such adversary implies the insecurity of the scheme.

The distinguishing attack on XCBv2 that we show here consists of generating the same counter (value of the variable S in the encryption procedure) for two different messages.

We assume an adversary which makes two encryption queries $(T^{(1)}, P^{(1)})$ and $(T^{(2)}, P^{(2)})$, where $T^{(1)} = T^{(2)} = T \in \mathcal{T}$ is an arbitrary tweak and $P^{(1)} = 0^{2n+i}$, $P^{(2)} = 0^{3n}$, for any $1 \leq i \leq (n-1)$. Note, in the standard it is required that $n = 128$ and message lengths should be multiples of 8 bits. One can select any specific i satisfying this condition. As a response to these queries the adversary gets back $C^{(1)}$ and $C^{(2)}$. If the first n bits of $C^{(1)}$ and $C^{(2)}$ are equal then the adversary concludes that it's oracle is that of XCBv2 and otherwise it concludes that it's oracle is a random permutation.

Now, we explain why this attack works. For the first query $(T^{(1)}, P^{(1)})$, the internal variable $S^{(1)}$ (see line 107) would be,

$$\begin{aligned} S^{(1)} &= E_{K_e}(0^n) \oplus H_h(0^n||T, 0^n||\mathsf{pad}(0^i)||0^n) \\ &= E_{K_e}(0^n) \oplus H_h(0^n||T, 0^{3n}). \end{aligned}$$

The first block of ciphertext for this query would be

$$C_1^{(1)} = 0^n \oplus E_{K_c}(\mathsf{inc}^0(S^{(1)})). \tag{3}$$

For the second query the variable $S^{(2)}$ would be

$$\begin{aligned} S^{(2)} &= E_{K_e}(0^n) \oplus H_h(0^n||T, 0^{2n}||0^n) \\ &= E_{K_e}(0^n) \oplus H_h(0^n||T, 0^{3n}). \end{aligned}$$

[4] In [12], the authors use a non-standard terminology. They do not distinguish between a pseudorandom permutation (PRP) and a strong pseudorandom permutation (SPRP). According to their definitions a PRP is what is generally understood as a SPRP.

Note that for computing $S^{(2)}$ no padding is required, as the plaintext is a multiple of n. Similarly, we have the first block of the ciphertext for the second query as

$$C_1^{(2)} = 0^n \oplus E_{K_c}(\mathsf{inc}^0(S^{(2)})). \tag{4}$$

Now, as $S^{(1)} = S^{(2)}$, hence it would always be the case that $C_1^{(1)} = C_1^{(2)}$. Thus the advantage of this adversary (who asks just two queries) is $1 - \frac{1}{2^n}$ which contradicts the bound given in Theorem 1.

We can have a general description of this attack. Let us consider two messages $P^{(1)}$ and $P^{(2)}$ as

$$P^{(1)} = P_1||\ldots||P_{m-1}||P_m,$$
$$P^{(2)} = P_1||\ldots||P_{m-1}||0^{n-p}||P_m.$$

Where $|P_{m-1}| = p < n$ and $|P_i| = n$ for $1 \leq i \leq m$ and $i \neq m-1$. Note, here the length of $P^{(2)}$, is a multiple of the block length n, which is not the case for $P^{(1)}$. Any two messages of this form will result in the same value of S, and hence the first $(m-2)$ blocks of ciphertext produced by this pair of different messages with the same tweak would be equal.

We can vary the size of P_{m-1} and obtain many different messages which will give the same value S. In [1], where XCBv2 has been standardized, the length in bits of the message is restricted to be a multiple of 8 bits, it is easy to see that with this restriction also our distinguishing attack works.

5.1 Some Comments About the Attack

1. The attack works because of the way the padding is applied. As the hash function takes as input the zero padded message, hence the length of the original message has no effect on the value of the first hash.
2. An easy way to bypass this attack is to restrict the message space to contain only messages whose lengths are multiples of the block length. In such messages, explicit padding would be not required and hence this attack does not work in case of XCBv2.
3. The attack does not depend on the padding function. If any other deterministic padding function at the block level is applied (say instead of padding zeros, one pads ones to make an incomplete block full) this attack can be easily modified to make it work. On the other hand, if a message level padding function is applied, say the given message is padded with 10^*, this would prevent the attack. This padding ensures injectivity so that distinct messages will map to distinct padded messages. On the other hand, this 10* padding scheme will mean that full block messages gets extended by one block. Since for disk encryption applications, messages will usually consist of full blocks, this increase by an extra block is undesirable. Maybe this is the reason why the designers did not opt for the 10* padding (and ended up with an insecure scheme).

6 Counterexample to Theorem 2: Collisions in the Increment Function

In the previous Section we showed a distinguishing attack which shows that the main security Theorem (Theorem 1) of XCBv2 is false. We also mentioned that if we restrict ourselves to only messages whose lengths are multiples of the block length of the block cipher then our distinguishing attack does not work. Here we show that even if we restrict ourselves to messages whose length are multiples of the block length, then too the proof of Theorem 1 as provided in [12] is not correct.

The proof of Theorem 1 as provided in [12] depends on the result stated in Theorem 2. We show that this result is also false by giving some counterexamples.

The result in Theorem 2 has also been used to argue about the security of an authenticated encryption scheme GCM. Recently, Iwata, Ohashi and Minematsu [9] pointed out an error in the security bound of GCM, this error also stems from the falsity of Theorem 2. The counter example that we present here is highly motivated by [9], but our examples are different.

For the discussion that follows, we shall treat a n-bit string $a = (a_{n-1}, a_{n-2}, \ldots, a_1, a_0)$ as a polynomial $a(x) = a_{n-1} \oplus a_{n-2}x \oplus \cdots \oplus a_1 x^{n-2} \oplus a_0 x^{n-1}$.

Let us consider $D = D' = T = T' = 0^{128}, P = 0^{384}$ and $P' = 0^{640}$. Then, according to Theorem 2

$$p_s = \Pr[h \xleftarrow{\$} \{0,1\}^{128} : H_h(T,P) = \mathsf{inc}^s(H_h(T,P'))] \leq \frac{8}{2^{128}},$$

for any index s. We show that $p_1 \geq 16/2^{128}$, $p_2 \geq 16/2^{128}$ and $p_4 \geq 15/2^{128}$, thus invalidating the theorem.

For our choice of P, P', T, T' and the description of the hash function in Eq. (1) we have $H_h(T,P) = L_1 h$ and $H_h(T',P') = L_2 h$, where $L_1 = x^{56} + x^{119} + x^{120}$ and $L_2 = x^{56} + x^{118} + x^{120}$. We were able to find 16 distinct values of h which satisfies

$$L_1 h \oplus \mathsf{inc}(L_2 h) = 0^{128}. \tag{5}$$

Also we found 16 distinct values of h satisfying $L_1 h \oplus \mathsf{inc}^2(L_2 h) = 0^{128}$, and 15 distinct values of h satisfying $L_1 h \oplus \mathsf{inc}^4(L_2 h) = 0^{128}$. These values are listed in Tables 1, 2 and 3. This suggests that $p_1 \geq 16/2^{128}$, $p_2 \geq 16/2^{128}$ and $p_4 \geq 15/2^{128}$.

0xBE7FFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0xBF7FFFFFFFFFFFFFFFFFFFFFFFFFFFFF
0xBB7FFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0xAB7FFFFFFFFFFFFFFFFFFFFFFFFFFFFF
0xEB7FFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0x297FFFFFFFFFFFFFFFFFFFFFFFFFFFFE
0xA57FFFFFFFFFFFFFFFFFFFFFFFFFFFF8	0xD37FFFFFFFFFFFFFFFFFFFFFFFFFFFE3
0xC97FFFFFFFFFFFFFFFFFFFFFFFFFFF8E	0xA17FFFFFFFFFFFFFFFFFFFFFFFFFFE3A
0xC37FFFFFFFFFFFFFFFFFFFFFFFFFF8EB	0x897FFFFFFFFFFFFFFFFFFFFFFFFFE3AE
0x637FFFFFFFFFFFFFFFFFFFFFFFFF8EBB	0x4F7FFFFFFFFFFFFFFFFFFFFFFFE3AED
0xFF7FFFFFFFFFFFFFFFFFFFFFFF8EBB5	0x797FFFFFFFFFFFFFFFFFFFFFE3AED6

Table 1. List of solutions for $L_1 h \oplus \mathsf{inc}(L_2 h) = 0^{128}$.

0xBEFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0xBCFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
0xB4FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0x94FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
0x14FFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0x52FFFFFFFFFFFFFFFFFFFFFFFFFFFFFC
0x88FFFFFFFFFFFFFFFFFFFFFFFFFFFFF1	0x64FFFFFFFFFFFFFFFFFFFFFFFFFFFFC7
0x50FFFFFFFFFFFFFFFFFFFFFFFFFFFF1D	0x80FFFFFFFFFFFFFFFFFFFFFFFFFFFC75
0x44FFFFFFFFFFFFFFFFFFFFFFFFFFF1D7	0xD0FFFFFFFFFFFFFFFFFFFFFFFFFFC75D
0xC6FFFFFFFFFFFFFFFFFFFFFFFFFF1D76	0x9EFFFFFFFFFFFFFFFFFFFFFFFFFC75DA
0x3CFFFFFFFFFFFFFFFFFFFFFFFFFF1D76B	0xF2FFFFFFFFFFFFFFFFFFFFFFFFFC75DAC

Table 2. List of solutions for $L_1 h \oplus \mathsf{inc}^2(L_2 h) = 0^{128}$.

0xBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0xBBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
0xABFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	0xEBFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
0x29FFFFFFFFFFFFFFFFFFFFFFFFFFFFFE	0xA5FFFFFFFFFFFFFFFFFFFFFFFFFFFFF8
0xD3FFFFFFFFFFFFFFFFFFFFFFFFFFFFE3	0xC9FFFFFFFFFFFFFFFFFFFFFFFFFFFF8E
0xA1FFFFFFFFFFFFFFFFFFFFFFFFFFFE3A	0xC3FFFFFFFFFFFFFFFFFFFFFFFFFFF8EB
0x89FFFFFFFFFFFFFFFFFFFFFFFFFFE3AE	0x63FFFFFFFFFFFFFFFFFFFFFFFFFF8EBB
0x4FFFFFFFFFFFFFFFFFFFFFFFFFE3AED	0xFFFFFFFFFFFFFFFFFFFFFFFFFF8EBB5
0x79FFFFFFFFFFFFFFFFFFFFFFFE3AED6	

Table 3. List of solutions for $L_1 h \oplus \mathsf{inc}^4(L_2 h) = 0^{128}$.

Next, we describe the method we used to find the solutions for Eq. (5). Similar methods can be used to find solutions for $L_1 h \oplus \mathsf{inc}^2(L_2 h) = 0^{128}$ and $L_1 h \oplus \mathsf{inc}^4(L_2 h) = 0^{128}$.

Our method is based on the following simple observation.

Definition 1. *For $a \in \{0,1\}^n$, i.e, $a(x) = a_{127} \oplus a_{126}x \oplus a_{125}x^2 \oplus \cdots \oplus a_1 x^{126} \oplus a_0 x^{127}$, where each $a_i \in \{0,1\}$. Define, $\mathsf{lbit}(a) = 127 - j$, where j is the smallest integer such that $0 \leq j \leq 127$, and $a_j = 0$. If such a j does not exist (such a j will not exist if all bits of a are 1) or $j > 31$, then we fix $\mathsf{lbit}(a) = 96$.*

Proposition 1. *Fix $a \in \{0,1\}^n$, if $\mathsf{lbit}(a) = j$, then $\mathsf{inc}(a) = a(x) \oplus x^j \oplus x^{j+1} \oplus \cdots \oplus x^{127}$.*

If we assume that a solution h of Eq. (5) is such that $\mathsf{lbit}(L_2 h) = j$, then by Proposition 1 we have

$$h = \frac{x^j \oplus x^{j+1} \oplus \cdots \oplus x^{127}}{L_1 \oplus L_2}. \tag{6}$$

Using $96 \leq j \leq 127$, in Eq. (6) we obtain 32 values of h. We substitute these values in eq. (5) to check whether the value obtained is really a solution. It turned out that sixteen of the 32 values obtained satisfied equation (5), and these values are reported in Table 1. It is easy to characterize $\mathsf{inc}^2(a)$ and $\mathsf{inc}^4(a)$ along the lines of Proposition 1, and using these characterizations we generate the data for Tables 2 and 3.

7 New Security Bound for XCBv2 on Full Block Messages

In this section we derive the information theoretic security bound for XCBv2 with the message space restricted to those messages which are multiples of the block length. We denote this version of XCBv2 as XCBv2fb. The security theorem applies only to XCBv2fb. As shown earlier XCBv2 is insecure.

For deriving this bound we replace the block cipher calls $E_{K_c}(), E_{K_d}(), E_{K_e}()$ by three permutations π_1, π_2, π_3 chosen uniformly at random from $\mathrm{Perm}(n)$. We also choose the hash key h uniformly at random from $\{0,1\}^n$. We call this construction as XCBv2fb[3Perm(n)]. The following theorem states the information theoretic security bound for XCBv2fb[3Perm(n)].

Theorem 3. *Consider an arbitrary adversary A which queries only with messages/ciphers whose lengths are multiples of n, and A asks a total of q queries of query complexity σ, where each query is at most ℓ blocks long. Then,*

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{XCBv2fb[3Perm}(n)]}(A) \leq \frac{(5 + 2^{22})\ell q \sigma}{2^n}. \tag{7}$$

7.1 Proof of Theorem 3

For proving (7), we need to consider an adversary's advantage in distinguishing XCBv2fb[3Perm(n)] from an oracle which simply returns random bit strings. This advantage is defined in the following manner.

$$\begin{aligned}\mathbf{Adv}^{\pm\mathrm{rnd}}_{\mathrm{XCBv2fb[3Perm}(n)]}(A) = \Big|&\Pr\left[\pi \xleftarrow{\$} \mathrm{Perm}(n) : A^{\mathbf{E}_\pi, \mathbf{D}_\pi} \Rightarrow 1\right] \\ &- \Pr\left[A^{\$(.,.),\$(.,.)} \Rightarrow 1\right]\Big|\end{aligned} \tag{8}$$

where $\$(.,M)$ or $\$(.,C)$ returns independently distributed random bits of length $|M|$ or $|C|$ respectively. The basic idea of proving (7) is as follows.

$$\begin{aligned}
\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A) &= \Big(\Pr\left[\pi \xleftarrow{\$} \mathrm{Perm}(n) : A^{\mathbf{E}_\pi,\mathbf{D}_\pi} \Rightarrow 1\right] \\
&\quad - \Pr\left[\boldsymbol{\pi} \xleftarrow{\$} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M}) : A^{\boldsymbol{\pi}(.,.),\boldsymbol{\pi}^{-1}(.,.)} \Rightarrow 1\right]\Big) \\
&= \Big(\Pr\left[\pi \xleftarrow{\$} \mathrm{Perm}(n) : A^{\mathbf{E}_\pi,\mathbf{D}_\pi} \Rightarrow 1\right] \\
&\quad - \Pr\left[A^{\$(.,.),\$(.,.)} \Rightarrow 1\right]\Big) \\
&\quad + \Big(\Pr\left[A^{\$(.,.),\$(.,.)} \Rightarrow 1\right] \\
&\quad - \Pr\left[\boldsymbol{\pi} \xleftarrow{\$} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M}) : A^{\boldsymbol{\pi}(.,.),\boldsymbol{\pi}^{-1}(.,.)} \Rightarrow 1\right]\Big) \\
&\leq \mathbf{Adv}^{\pm\mathrm{rnd}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A) + \binom{q}{2}\frac{1}{2^n} \qquad (9)
\end{aligned}$$

where q is the number of queries made by the adversary. For a proof of the last inequality see [8]. Thus, the main task of the proof now reduces to obtaining an upper bound on $\mathbf{Adv}^{\pm\mathrm{rnd}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A)$.

We prove this by the usual technique of sequence of games. The games XCB1, RAND1 are described in Fig. 2 and game RAND2 is described in Fig. 3.

Game XCB1 described in Fig. 2 is just a rewrite of the algorithm of XCBv2fb, but it uses three independent random permutations π_1, π_2, π_3 instead of the block cipher calls. The permutations are constructed on the fly. If we denote the encryption scheme and decryption scheme of XCBv2fb[3Perm(n)] by $\mathbf{E}_{\pi_1,\pi_2,\pi_3}$ and $\mathbf{D}_{\pi_1,\pi_2,\pi_3}$ respectively then, by our choice of notation, we can write

$$\Pr[A^{\mathbf{E}_{\pi_1,\pi_2,\pi_3},\mathbf{D}_{\pi_1,\pi_2,\pi_3}} \Rightarrow 1] = \Pr[A^{\mathrm{XCB1}} \Rightarrow 1]. \qquad (10)$$

Game RAND1 is also described in Fig. 2 with the boxed entries removed. In this game it is not guaranteed that π_i ($i = 1, 2, 3$) are permutations as though we do the consistency checks but we do not reset the values of Y (in Ch-π_i) and X (in Ch-π_i^{-1}). Thus, the games XCB1 and RAND1 are identical apart from what happens when the bad flag is set. By the fundamental lemma of game-playing, we have

$$|\Pr[A^{\mathrm{XCB1}} \Rightarrow 1] - \Pr[A^{\mathrm{RAND1}} \Rightarrow 1]| \leq \Pr[A^{\mathrm{RAND1}} \text{ sets } \mathsf{bad}] \qquad (11)$$

Another important thing to note is that in RAND1 the adversary gets random strings in response to both its encryption and decryption queries. Hence,

$$\Pr[A^{\mathrm{RAND1}} \Rightarrow 1] = \Pr[A^{\$(.,.),\$(.,.)} \Rightarrow 1] \qquad (12)$$

So using the definition of $\mathbf{Adv}^{\pm\mathrm{rnd}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A)$ and Eqs. (11) and (12) we get

$$\begin{aligned}
\mathbf{Adv}^{\pm\mathrm{rnd}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A) &= |\Pr[A^{\mathbf{E}_{\pi_1,\pi_2,\pi_3},\mathbf{D}_{\pi_1,\pi_2,\pi_3}} \Rightarrow 1] \\
&\quad - \Pr[A^{\$(.,.),\$(.,.)} \Rightarrow 1]| \qquad (13) \\
&= |\Pr[A^{\mathrm{XCB1}} \Rightarrow 1] \\
&\quad - \Pr[A^{\mathrm{RAND1}} \Rightarrow 1]| \\
&\leq \Pr[A^{\mathrm{RAND1}} \text{ sets } \mathsf{bad}] \qquad (14)
\end{aligned}$$

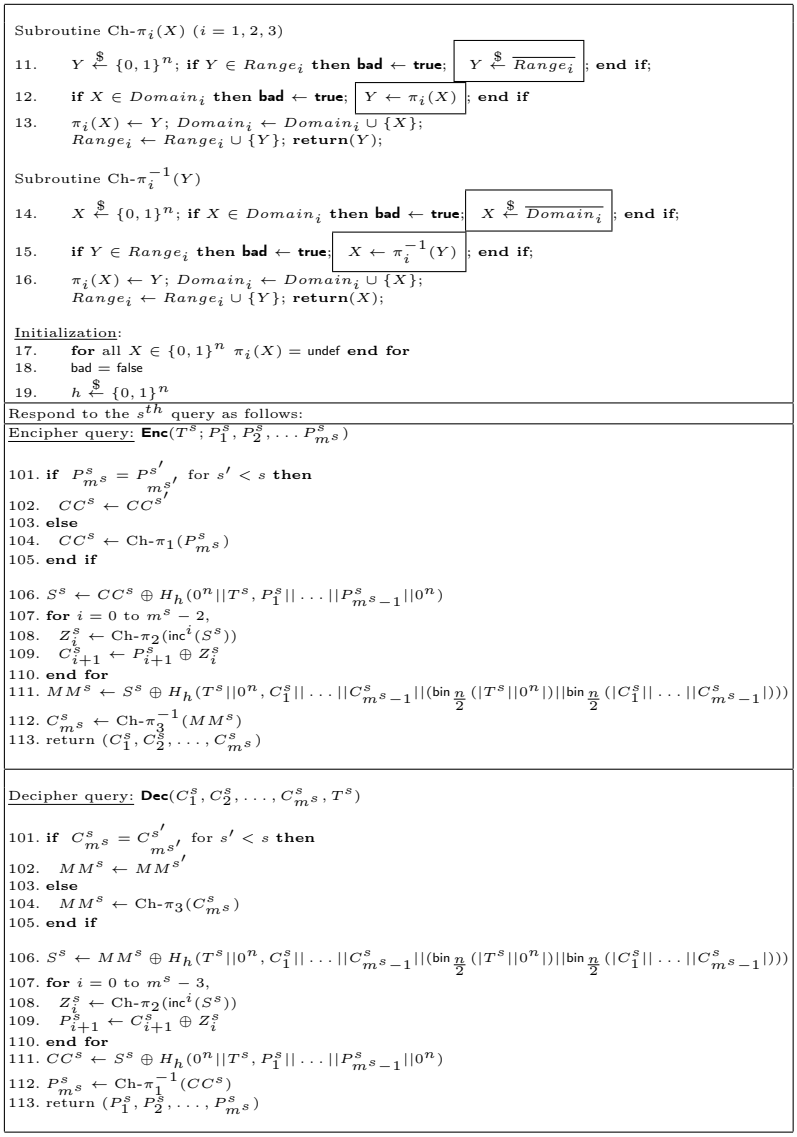

Fig. 2. Games XCB1 and RAND1: In RAND1 the boxed entries are removed.

Respond to the s^{th} adversary query as follows:

ENCIPHER QUERY $\mathbf{Enc}(T^s; P_1^s, P_2^s, \ldots, P_{m^s}^s)$

11. $ty^s = \mathsf{Enc}$
12. $C_1^s||C_2^s||\ldots||C_{m^s-1}^s||C_{m^s}^s \xleftarrow{\$} \{0,1\}^{nm^s}$
13. **return** $C_1^s||C_2^s||\ldots||C_{m^s}^s$

DECIPHER QUERY $\mathsf{Dec}(T^s; C_1^s, C_2^s, \ldots, C_{m^s}^s)$

21. $ty^s = \mathsf{Dec}$
22. $P_1^s||P_2^s||\ldots||P_{m_s-1}^s||P_{m^s}^s \xleftarrow{\$} \{0,1\}^{nm^s}$
23. **return** $P_1^s||P_2^s||\ldots||P_{m^s}^s$

Finalization:

$h \xleftarrow{\$} \{0,1\}^n$

Case $ty^s = \mathsf{Enc}$:

101. **if** $P_{m^s}^s = P_{m^{s'}}^{s'}$ for $s' < s$ **then**
102. $CC^s \leftarrow CC^{s'}$
103. **else**
104. $CC^s \xleftarrow{\$} \{0,1\}^n$
105. $\mathcal{D}_1 \leftarrow \mathcal{D}_1 \cup \{P_{m^s}^s\}$
106. $\mathcal{R}_1 \leftarrow \mathcal{R}_1 \cup \{CC^s\}$
107. **end if**
108. $S^s \leftarrow CC^s \oplus H_h(0^n||T^s, P_1^s||\ldots||P_{m^s-1}^s||0^n)$
109. $MM^s \leftarrow S^s \oplus H_h(T^s||0^n, C_1^s||\ldots||C_{m^s-1}^s||(\mathsf{bin}\,\frac{n}{2}\,(|T^s||0^n|)||\mathsf{bin}\,\frac{n}{2}\,(|C_1^s||\ldots||C_{m^s-1}^s|)))$
110. $\mathcal{D}_3 \leftarrow \mathcal{D}_3 \cup \{C_{m^s}^s\}$
111. $\mathcal{R}_3 \leftarrow \mathcal{R}_3 \cup \{MM^s\}$
112. **for** $i = 0$ to $m^s - 2$,
113. $Y_i^s \leftarrow C_{i+1}^s \oplus P_{i+1}^s$
114. $\mathcal{D}_2 \leftarrow \mathcal{D}_2 \cup \{\mathsf{inc}^i(S^s)\}$
115. $\mathcal{R}_2 \leftarrow \mathcal{R}_2 \cup \{Y_i^s\}$
116. **end for**

Case $ty^s = \mathsf{Dec}$:

101. **if** $C_{m^s}^s = C_{m^{s'}}^{s'}$ for $s' < s$ **then**
102. $MM^s \leftarrow MM^{s'}$
103. **else**
104. $MM^s \xleftarrow{\$} \{0,1\}^n$
105. $\mathcal{D}_3 \leftarrow \mathcal{D}_3 \cup \{C_{m^s}^s\}$
106. $\mathcal{R}_3 \leftarrow \mathcal{R}_3 \cup \{MM^s\}$
107. **end if**
108. $S^s \leftarrow MM^s \oplus H_h(T^s||0^n, C_1^s||\ldots||C_{m^s-1}^s||(\mathsf{bin}\,\frac{n}{2}\,(|T^s||0^n|)||\mathsf{bin}\,\frac{n}{2}\,(|C_1^s||\ldots||C_{m^s-1}^s|)))$
109. $CC^s \leftarrow S^s \oplus H_h(0^n||T^s, P_1^s||\ldots||P_{m^s-1}^s||0^n)$
110. $\mathcal{D}_1 \leftarrow \mathcal{D}_1 \cup \{P_{m^s}^s\}$
111. $\mathcal{R}_1 \leftarrow \mathcal{R}_1 \cup \{CC^s\}$
112. **for** $i = 0$ to $m^s - 2$,
113. $Y_i^s \leftarrow C_{i+2}^s \oplus P_{i+2}^s$
114. $\mathcal{D}_2 \leftarrow \mathcal{D}_2 \cup \{\mathsf{inc}^i(S^s)\}$
115. $\mathcal{R}_2 \leftarrow \mathcal{R}_2 \cup \{Y_i^s\}$
116. **end for**

SECOND PHASE

$\mathsf{bad} = \mathsf{false}$;
if (some value occurs more than once in $\mathcal{D}_i$, $i = 1, 2, 3$) **then** $\mathbf{bad} = \mathsf{true}$ **end if**;
if (some value occurs more than once in $\mathcal{R}_i$, $i = 1, 2, 3$) **then** $\mathbf{bad} = \mathsf{true}$ **end if**.

Fig. 3. Game RAND2

Game RAND2 is syntactically different from RAND1 by the fact that here the permutations are no more maintained, and in response to an encryption/decryption query of an adversary, random strings of appropriate lengths are immediately returned. Later, in the finalization step of the game the internal variables are adjusted and the appropriate variables are inserted in the multi sets $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ and $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$. In the last step of finalization, it is checked whether there is a collision in the multi sets $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$, $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$. If a collision occurs, then the bad flag is set.
Games RAND1 and RAND2 are indistinguishable to the adversary, as in both cases it gets random strings in response to its queries. Also, the probability with which RAND1 sets bad is same as the probability with which RAND2 sets bad. Thus we get:

$$\Pr[A^{\mathrm{RAND1}} \text{ sets } \mathsf{bad}] = \Pr[A^{\mathrm{RAND2}} \text{ sets } \mathsf{bad}] \tag{15}$$

Thus from equations (14) and (15) we obtain

$$\mathbf{Adv}^{\pm\mathrm{rnd}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A) \leq \Pr[A^{\mathrm{RAND2}} \text{ sets } \mathsf{bad}] \tag{16}$$

Now our goal is to bound $\Pr[A^{\mathrm{RAND2}} \text{ sets } \mathsf{bad}]$. We notice that in Game RAND2 the bad flag is set when there is a collision in either of the sets $\mathcal{D}_i$ or $\mathcal{R}_i$. So if COLLD_i and COLLR_i denote the events of a collision in $\mathcal{D}_i$ and $\mathcal{R}_i$ respectively then we have

$$\Pr[A^{\mathrm{RAND2}} \text{ sets } \mathsf{bad}] \leq \sum_{1 \leq i \leq 3} \left(\Pr[\mathsf{COLLR}_i] + \Pr[\mathsf{COLLD}_i]\right). \tag{17}$$

In the next section we compute bounds on $\Pr[\mathsf{COLLR}_i]$ and $\Pr[\mathsf{COLLD}_i]$.

7.2 Collision Analysis

Before we go into the collision analysis we summarize some useful results. Firstly we concentrate on the following problem which was stated in [9].

Problem 1. For $0 \leq r \leq 2^{32} - 1$, consider the set

$$\mathbb{Y}_r = \left\{\mathsf{bin}_{32}\left(\mathsf{int}(Y) + r \bmod 2^{32}\right) \oplus Y : Y \in \{0,1\}^{32}\right\}.$$

Let α_r denote the cardinality of $\mathbb{Y}_r$. Find α_r.

In [9], α_r was described using two recurrence relations, and numerical values of α_r was computed for all $0 \leq r \leq 2^{32} - 1$. It was found that

$$\alpha_{max} = \max_{0 \leq r \leq 2^{32}-1} \{\alpha_r\} = 3524578 < 2^{22}. \tag{18}$$

Next, we state a result (similar to Lemma 2 in [9]) which would be central to the collision analysis that we present here. Also this lemma would be used later in the security proof of XCBv1.

Lemma 1. *1. Let $X, Y, X', Y' \in \{0,1\}^*$, such that $(X, Y) \neq (X', Y')$. Let $C, C' \in \{0,1\}^n$ and $h \xleftarrow{\$} \{0,1\}^n$, $S = C \oplus H_h(X, Y)$, and $S' = C' \oplus H_h(X', Y')$, where $H_h(.)$ is defined in Eq. (1). Then, for any $0 \leq r \leq 2^{32}-1$,*

$$\Pr[\mathsf{inc}^r(S) \oplus S' = 0] \leq \frac{\max\{\ell^s, \ell^{s'}\}\alpha_r}{2^n}$$

2. Let $X, Y, X', Y' \in \{0,1\}^*$, $C, C' \in \{0,1\}^n$, $h_1, h_2 \xleftarrow{\$} \{0,1\}^n$, $S = C \oplus H_{h_1}(X, Y)$, and $S' = C' \oplus H_{h_2}(X', Y')$. Then, for any $0 \leq r \leq 2^{32} - 1$,

$$\Pr[\mathsf{inc}^r(S) \oplus S' = 0] \leq \frac{\max\{\ell^s, \ell^{s'}\}\alpha_r}{2^n}.$$

In both cases ℓ^s and $\ell^{s'}$ are the degrees of the two polynomials S and S' respectively. In the first case, the probability is taken over the random choice of h, and in the second case it is taken over the random choice of h_1, h_2.

Proof. The proof follows the same idea as the proof of Lemma 2 in [9]. For $0 \leq r \leq 2^{32}-1$, define $f_r : \{0,1\}^{32} \to \{0,1\}^{32}$ as

$$f_r(Z) = \mathsf{bin}_{32}\left(\mathsf{int}(Z) + r \bmod 2^{32}\right).$$

Let, $\mathbb{Y}_r = \{Y_1, Y_2, \ldots, Y_{\alpha_r}\}$, and define

$$\mathcal{Y}_j = \{Z \in \{0,1\}^{32} : f_r(Z) \oplus Z = Y_j\}.$$

It is easy to see that $\mathcal{Y}_1 \cup \mathcal{Y}_2 \cup \cdots \cup \mathcal{Y}_{\alpha_r} = \{0,1\}^{32}$ and for $j \neq j'$, $\mathcal{Y}_j \cap \mathcal{Y}_{j'} = \emptyset$.

Also if $Z \in \mathcal{Y}_j$, then $\mathsf{bin}_{32}\left(\mathsf{int}(Z) + r \bmod 2^{32}\right) = Z \oplus Y_j$. For $0 \leq j \leq \alpha_r$ define the event $\mathbf{E}_j$ as

$$\mathbf{E}_j \equiv (\mathsf{inc}^r(S) \oplus S' = 0) \wedge (\mathsf{lsb}_{32}(S) \in \mathcal{Y}_j).$$

Hence, we have

$$\Pr[\mathsf{inc}^r(S) \oplus S' = 0] = \sum_{j=1}^{\alpha_r} \Pr[\mathbf{E}_j]. \tag{19}$$

Further, $\Pr[\mathbf{E}_j] = \Pr[\mathsf{inc}^r(S) \oplus S' = 0 | \mathsf{lsb}_{32}(S) \in \mathcal{Y}_j] \Pr[\mathsf{lsb}_{32}(S) \in \mathcal{Y}_j]$. Now, if $\mathsf{lsb}_{32}(S) \in \mathcal{Y}_j$, then $\mathsf{inc}^r(S) = S \oplus (0^{n-32}||Y_j)$. Hence,

$$\begin{aligned}\Pr[\mathsf{inc}^r(S) \oplus S' = 0 | \mathsf{lsb}_{32}(S) \in \mathcal{Y}_j] &= \Pr[S \oplus (0^{n-32}||Y_j) \oplus S' = 0 | \mathsf{lsb}_{32}(S) \in \mathcal{Y}_j] \\ &= \Pr\left[S \oplus S' \oplus D_j = 0 | \mathsf{lsb}_{32}(S) \in \mathcal{Y}_j\right], \end{aligned} \tag{20}$$

where $D_j = (0^{n-32}||Y_j)$ is a non-random quantity. So,

$$\begin{aligned}\Pr[\mathsf{inc}^r(S) \oplus S' = 0] &= \sum_{j=1}^{\alpha_r} \Pr[\mathbf{E}_j] \\ &= \sum_{j=1}^{\alpha_r} \Pr\left[S \oplus S' \oplus D_j = 0 | \mathsf{lsb}_{32}(S) \in \mathcal{Y}_j\right] \Pr[\mathsf{lsb}_{32}(S) \in \mathcal{Y}_j] &(21)\\ &= \sum_{j=1}^{\alpha_r} \Pr\left[(S \oplus S' \oplus D_j = 0) \wedge (\mathsf{lsb}_{32}(S) \in \mathcal{Y}_j)\right] &(22)\\ &\leq \sum_{j=1}^{\alpha_r} \Pr\left[S \oplus S' \oplus D_j = 0\right]. &(23)\end{aligned}$$

Now, we consider the two cases of the lemma.

Case 1: h is selected uniformly at random from $\{0,1\}^n$, and $S = H_h(X,Y)$ and $S' = H_h(X',Y')$. In this case

$$\Pr[S \oplus S' \oplus D_j = 0] \leq \frac{\max\{\ell^s, \ell^{s'}\}}{2^n}, \tag{24}$$

as $S \oplus S' \oplus D_j$ is a nonzero polynomial in h of degree at most $\max\{\ell^s, \ell^{s'}\}$.

Case 2: h_1, h_2 are selected independently and uniformly at random from $\{0,1\}^n$, and $S = H_{h_1}(X,Y)$ and $S' = H_{h_2}(X',Y')$. Note, according to the definition of $H(.)$, both S and S' are non-zero polynomials. Hence, $S \oplus S' \oplus D_j$ is a non-zero multivariate polynomial on (h_1, h_2) of (total) degree at most $\max\{\ell^s, \ell^{s'}\}$. Hence, by the Schwartz-Zippel lemma

$$\Pr[S \oplus S' \oplus D_j = 0] \leq \frac{\max\{\ell^s, \ell^{s'}\}}{2^n}. \tag{25}$$

In both cases, we get $\Pr[S \oplus S' \oplus D_j = 0] \leq (\max\{\ell^s, \ell^{s'}\})/2^n$. So, Eq. (23) becomes

$$\Pr[\mathsf{inc}^r(S) \oplus S' = 0] = \frac{\alpha_r \max\{\ell^s, \ell^{s'}\}}{2^n}.$$

□

Remark: In moving from (21) to (22) an inequality is involved. It is not clear whether this inequality is tight, i.e., whether there is a choice of (X, Y) and (X', Y') for which the upper bound is attained. Further, it is also not clear how to analyze the probability in (21) which will show that the upper bound is tight (possibly up to a small constant) or to prove a significantly lower upper bound. In such a scenario, for concrete security analysis it is appropriate to proceed with the upper bound that can indeed be proved. We note that the approximation involved in moving from (21) to (22) is also used in Lemma 2 of [9]. Accordingly, the issue of tightness of the upper bound discussed above also applies to the upper bound obtained in [9] for GCM.

In the rest of the section we analyze the collision probabilities in the sets $\mathcal{D}_i$ and $\mathcal{R}_i$. After q queries of the adversary where the s^{th} query has m^s blocks of plaintext or ciphertext and t^s blocks of tweak, then the sets $\mathcal{D}_i$ and $\mathcal{R}_i$ can be written as follows:

$\mathcal{D}_1 = \{P^s_{m^s} : 1 \leq s \leq q\}$
$\mathcal{D}_2 = \bigcup_{s=1}^{q} \{\mathsf{inc}^j(S^s) : 0 \leq j \leq m^s - 2\}$
$\mathcal{D}_3 = \{C^s_{m^s} : 1 \leq s \leq q\}$

$\mathcal{R}_1 = \{CC^s : 1 \leq s \leq q\}$
$\mathcal{R}_2 = \bigcup_{s=1}^{q} \{Y^s_j = C^s_{j+1} \oplus P^s_{j+1} : 0 \leq j \leq m^s - 2\}$
$\mathcal{R}_3 = \{MM^s : 1 \leq s \leq q\}$

Noting the following points will help in the following analysis:

1. For the s-th query $ty^s \in \{enc, dec\}$ will denote whether the query is an encryption or a decryption query.
2. In each query, the adversary specifies a tweak T^s, we consider $t^s = \lceil |T^s|/n \rceil$. Thus, for any s, H_h either in line 108 or 109 of game RAND2 has degree at most $m^s + t^s + 2$. We denote $\sigma = \sum_s t^s + \sum_s m^s$. We denote $\ell^{s,s'} = \max\{t^s + m^s, t^{s'} + m^{s'}\} + 2$.
3. In game RAND2 the hash key h is selected uniformly at random from $\{0,1\}^n$, furthermore h is independent of all other variables.
4. As a response to each query, A receives either $(C^s_1, C^s_2, \ldots, C^s_{m^s})$ or $(P^s_1, P^s_2, \ldots, P^s_{m^s})$. These variables are independent of h, thus the queries made by A are also independent of h.
5. For an encryption query, the response received by A is $(C^s_1, C^s_2, \ldots, C^s_{m^s})$ and for a decryption query the response received is $(P^s_1, P^s_2, \ldots, P^s_{m^s})$. Both these responses are uniformly distributed and independent of other variables.

In the following claims we bound the required collision probabilities.

Claim 4. $\Pr[\mathsf{COLLD}_1] \leq \binom{q}{2}/2^n$

Proof. To compute $\Pr[P^s_{m^s} = P^{s'}_{m^{s'}}]$ there are the following two cases to consider.

Case I: $ty^s = ty^{s'} = enc$. In this case $\Pr[P^s_{m^s} = P^{s'}_{m^{s'}}] = 0$, because of the condition in line 101 of RAND2.
Case II: At least one of ty^s or $ty^{s'}$ is dec. Without loss of generality, if $ty^s = dec$, then $P^s_{m^s}$ is a uniform random n-bit string, hence $\Pr[P^s_{m^s} = P^{s'}_{m^{s'}}] = 1/2^n$.

As, there are q elements in $\mathcal{D}_1$, hence $\Pr[\mathsf{COLLD}_1] \leq \binom{q}{2}/2^n$.

□

Claim 5. $\Pr[\mathsf{COLLD}_2] \leq (\ell q \sigma)\alpha_{max}/2^n$

Proof. $\mathcal{D}_2 = D_1 \cup D_2 \cup \cdots \cup D_q$, where

$$D_s = \left\{\mathsf{inc}^j(S^s) : 0 \leq j \leq m^s - 2\right\}, \text{ for } 0 \leq s \leq q$$

and

$$S^s = \begin{cases} CC^s \oplus H_h(0^n||T^s, P_1^s||\cdots||P_{m^s-1}^s||0^n) & \text{if } ty^s = enc \\ MM^s \oplus H_h(0^n||T^s, C_1^s||\cdots||C_{m^s-1}^s||L) & \text{if } ty^s = dec, \end{cases}$$

where $L = (\mathsf{bin}_{\frac{n}{2}}(|T^s||0^n|)||\mathsf{bin}_{\frac{n}{2}}(|C_1^s||\ldots||C_{m^s-1}^s|))$.

It is easy to see that if $x_1, x_2 \in D_s$ then $\Pr[x_1 = x_2] = 0$. Thus we only need to bound collisions between x_1, x_2, such that $x_1 \in D_s$ and $x_2 \in D_{s'}$, for $s \neq s'$. For $s \neq s'$ define $\mathsf{COLL}(D_s, D_{s'})$ as the event that at least one element of D_s collides with one element of $D_{s'}$. Hence we have

$$\Pr[\mathsf{COLLD}_2] \leq \sum_{1 \leq s < s' \leq q} \Pr[\mathsf{COLL}(D_s, D_{s'})]. \tag{26}$$

Also it is easy to see that,

$$\mathsf{inc}^j(S^s) \oplus \mathsf{inc}^{j'}(S^{s'}) = \begin{cases} \mathsf{inc}^{j-j'}(S^s) \oplus S^{s'} \text{ if } j \geq j' \\ S^s \oplus \mathsf{inc}^{j'-j}(S^{s'}) \text{ if } j < j'. \end{cases}$$

Now, we define the following events

$$\mathsf{U}_i \equiv \mathsf{inc}^i(S^s) \oplus S^{s'} = 0, \quad \text{for } 0 \leq i \leq m^s - 2, \tag{27}$$

$$\mathsf{W}_i \equiv S^s \oplus \mathsf{inc}^i(S^{s'}) = 0, \quad \text{for } 0 \leq i \leq m^{s'} - 2. \tag{28}$$

Hence,

$$\mathsf{COLL}(D_s, D_{s'}) = \left(\bigcup_{i=0}^{m^s-2} \mathsf{U}_i\right) \bigcup \left(\bigcup_{i=0}^{m^{s'}-2} \mathsf{W}_i\right). \tag{29}$$

Now we will bound $\Pr[\mathsf{U}_i]$. We assume $s < s'$. For computing $\Pr[\mathsf{inc}^i(S^s) \oplus S^{s'} = 0]$, we have the following cases:

Case I: $ty^s = ty^{s'} = enc$ and $P_{m^s}^s \neq P_{m^{s'}}^{s'}$.
Case II: $ty^s = ty^{s'} = dec$ and $C_{m^s}^s \neq C_{m^{s'}}^{s'}$.
Case III: The rest.

For Case I, CC^s is chosen randomly hence for any i

$$\Pr[\mathsf{inc}^i(S^s) \oplus S^{s'} = 0] = \frac{1}{2^n}. \tag{30}$$

Similarly for Case II, MM^s is chosen randomly hence the same probability of Eq. (30) holds.

For case III note that both S^s and $S^{s'}$ are polynomials of h of degree at most $\ell^{s,s'} = \max\{m^s + t^s, m^{s'} + t^{s'}\} + 2$. Then using Lemma 1, we have

$$\Pr[\mathsf{inc}^i(S^s) \oplus S^{s'} = 0] \leq \frac{\alpha_{max}\ell^{s,s'}}{2^n}$$

Thus,

$$\Pr[\mathsf{U}_i] \leq \frac{\alpha_{max}\ell^{s,s'}}{2^n}, \text{ for all } 0 \leq i \leq m^s - 2. \tag{31}$$

By exactly the same argument we can show that

$$\Pr[\mathsf{W}_i] \leq \frac{\alpha_{max}\ell^{s,s'}}{2^n}, \text{ for all } 0 \leq i \leq m^{s'} - 2. \tag{32}$$

Thus using Eqs.(29), (31), (32), we have

$$\Pr[\mathsf{COLL}(D_s, D_{s'})] \leq \frac{\alpha_{max}(m^s + m^{s'} - 4)\ell^{s,s'}}{2^n}. \tag{33}$$

Using equations (33) and (26) we have

$$\begin{aligned}\Pr[\mathsf{COLLD}_2] &\leq \sum_{1\leq s<s'\leq q} \frac{\alpha_{max}(m^s+m^{s'}-4)\ell^{s,s'}}{2^n} \\ &\leq \frac{\ell\alpha_{max}}{2^n}\sum_{1\leq s<s'\leq q}(m^s+m^{s'}) \\ &\leq \frac{\ell\alpha_{max}q\sigma}{2^n}.\end{aligned}$$

□

Claim 6. $\Pr[\mathsf{COLLD}_3] \leq \binom{q}{2}/2^n$

The proof is similar to that of Claim 4.

Claim 7. $\Pr[\mathsf{COLLR}_1] \leq \frac{(q-1)\sigma}{2^n} + \frac{1}{2^n}\binom{q}{2}$

Proof. If $ty^s = enc$ then CC^s is selected uniformly at random from $\{0,1\}^n$. And if $ty^s = ty^{s'} = enc$ and $P^s_{m_s} = P^{s'}_{m_{s'}}$, then CC^s does not enter $\mathcal{R}_1$. Thus if $ty^s = enc$ or $ty^{s'} = enc$, then $\Pr[CC^s = CC^{s'}] = 1/2^n$.

We now need to settle the case when $ty^s = ty^{s'} = dec$.

If $ty^s = dec$ then

$$\begin{aligned}CC^s = MM^s \oplus H_h(0^n||T^s, P^s_1||\ldots||P^s_{m^s-1}||0^n) \\ \oplus H_h(T^s||0^n, C^s_1||\ldots||C^s_{m^s-1}||(\mathsf{bin}_{\frac{n}{2}}(|T^s||0^n|)||\mathsf{bin}_{\frac{n}{2}}(|C^s_1||\ldots||C^s_{m^s-1}|))).\end{aligned}$$

We assume $(T^s_1, T^s_2, \cdots, T^s_{p^s}) = \mathsf{parse}_n(T^s)$, and fix the following notations:

$$\begin{aligned}\mathsf{len}^s &= \mathsf{bin}_{\frac{n}{2}}(|T^s||0^n|)||\mathsf{bin}_{\frac{n}{2}}(|C^s_1||\ldots||C^s_{m^s-1}|), \\ \mathsf{Mix}(T^s) &= T^s_1||(T^s_1 \oplus T^s_2)||(T^s_2 \oplus T^s_3)||\cdots||(T_{p^s-1} \oplus T_{p^s})||T_{p^s}, \\ PC^s &= (P^s_1 \oplus C^s_1)||(P^s_2 \oplus C^s_2)||\cdots||(P^s_{m^s-1} \oplus C^s_{m^s-1}).\end{aligned}$$

Then, according to the definition of $H_h()$ in Eq. (1),

$$CC^s = MM^s \oplus H_h(\mathsf{Mix}(T^s), PC^s||\mathsf{len}^s).$$

We need to bound $\Pr[CC^s = CC^{s'}]$ where $ty^s = ty^{s'} = dec$. There are the following cases to consider.

Case 1: $\mathsf{len}^s \neq \mathsf{len}^{s'}$. There are two sub-cases to handle. If the degrees of $H_h(\mathsf{Mix}(T^s), PC^s||\mathsf{len}^s)$ and $H_h(\mathsf{Mix}(T^{s'}), PC^{s'}||\mathsf{len}^{s'})$ are different then $CC^s \oplus CC^{s'}$ is a nonzero polynomial of degree at most $\ell^{s,s'}$. If the degrees of the above polynomials are the same then in the polynomial $CC^s \oplus CC^{s'}$, a term with the coefficient $(\mathsf{len}^s \oplus \mathsf{len}^{s'})$ occurs, which means $CC^s \oplus CC^{s'}$ is a nonzero polynomial of degree at most $\ell^{s,s'}$. Thus, for both cases $\Pr[CC^s = CC^{s'}] \leq \ell^{s,s'}/2^n$.

Case 2: $\mathsf{len}^s = \mathsf{len}^{s'}$. Note, that this means $m^s = m^{s'}$, also $|T^s| = |T^{s'}|$.

Sub case 2a: $m^s = m^{s'} = 1$. In this case, two things can happen.

i. $C_{m^s} = C_{m^{s'}}$: It necessarily follows that $T^s \neq T^{s'}$, as otherwise the two queries will become identical. With $T^s \neq T^{s'}$, we have $\mathsf{Mix}(T^s) \neq \mathsf{Mix}(T^{s'})$, thus making $CC^s \oplus CC^{s'}$ a non-zero polynomial of degree at most $\ell^{s,s'}$, hence $\Pr[CC^s = CC^{s'}] \leq \ell^{s,s'}/2^n$.

ii. $C_{m^s} \neq C_{m^{s'}}$: In this case MM^s and $MM^{s'}$ are independent and uniformly distributed, from which it follows that C_{m^s} and $C_{m^{s'}}$ are also independent and uniformly distributed. Hence, $\Pr[CC^s = CC^{s'}] \leq 1/2^n$.

Sub case 2b: $m^s = m^{s'} \geq 2$. Here also we consider two different scenarios:

i. $\mathsf{Mix}(T^s) \neq \mathsf{Mix}(T^{s'})$: In this case, $CC^s \oplus CC^{s'}$ is a nonzero polynomial of degree at most $\ell^{s,s'}$, hence $\Pr[CC^s = CC^{s'}] \leq \ell^{s,s'}/2^n$.

ii. $\mathsf{Mix}(T^s) = \mathsf{Mix}(T^{s'})$: Let E_0 be the event that $PC^s = PC^{s'}$, i.e., $P_i^s \oplus C_i^s = P_i^{s'} \oplus C_i^{s'}$ for all $1 \leq i \leq m^s - 1$. As each P_i is selected uniformly and independently at random from $\{0,1\}^n$, thus irrespective of the choice of C_is, we have $\Pr[E_0] = 1/2^{(m^s-1)}$, and $\Pr[CC^s = CC^{s'}|\overline{\mathsf{E}_0}] \leq \ell^{s,s'}/2^n$.

Hence,

$$\begin{aligned}\Pr[CC^s = CC^{s'}] &= \Pr[CC^s = CC^{s'}|\mathsf{E}_0]\Pr[\mathsf{E}_0] + \Pr[CC^s = CC^{s'}|\overline{\mathsf{E}_0}]\Pr[\overline{\mathsf{E}_0}] \\ &\leq \frac{\ell^{s,s'}}{2^n} + \left(\frac{1}{2^n}\right)^{m^s-1} \\ &\leq \frac{\ell^{s,s'}}{2^n} + \frac{1}{2^n}.\end{aligned}$$

Hence we have

$$\begin{aligned}\Pr[\mathsf{COLLR}_1] &\leq \sum_{1\leq s<s'\leq q}\left(\frac{\ell^{s,s'}}{2^n} + \frac{1}{2^n}\right) \\ &\leq \frac{(q-1)\sigma}{2^n} + \frac{1}{2^n}\binom{q}{2} \qquad (34)\end{aligned}$$

□

Claim 8. $\Pr[\mathsf{COLLR}_2] \leq \binom{\sum_s m^s - q}{2}/2^n$

Proof. The elements in $\mathcal{R}_2$ are of the form $C_i^s \oplus P_i^s$, hence irrespective of s being an encryption or a decryption query each element in $\mathcal{R}_2$ is a uniform random element in $\{0,1\}^n$, moreover $\mathcal{R}_2$ contains $\sum_{s=1}^q m^s - q$ elements. Hence the claim follows.

□

Claim 9. $\Pr[\mathsf{COLLR}_3] \leq \frac{(q-1)\sigma}{2^n} + \frac{1}{2^n}\binom{q}{2}$.

The proof is similar to that of Claim 7.

Using Claims 4 to 9 and equations (16) and (17), we have

$$\begin{aligned}\mathbf{Adv}^{\pm\mathrm{rnd}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A) &\leq \frac{4}{2^n}\binom{q}{2} + \frac{\ell q\sigma\alpha_{max}}{2^n} + \frac{1}{2^n}\binom{\sum_s m^s - q}{2} + \frac{2\sigma(q-1)}{2^n} \\ &\leq \frac{4\sigma^2}{2^n} + \frac{\alpha_{max}\ell q\sigma}{2^n}. \qquad (35)\end{aligned}$$

Now, using equations (9), (18) and (35) we have

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{XCBv2fb}[3\mathrm{Perm}(n)]}(A) \leq \frac{(5+2^{22})\ell q\sigma}{2^n}$$

as desired. □

8 Security of XCBv1

Till date the security of XCBv1 has been informally argued. We provide a formal security argument for XCBv1 which shows security for arbitrary length messages and provides a concrete security bound. The following theorem specifies the security for XCBv1.

Theorem 10. *Consider an arbitrary adversary A with query complexity σ, each query of A is at most ℓ blocks long, and contains a message/cipher at least n bits long. Then,*

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{XCBv1}[3\mathrm{Perm}(n)]}(A) \leq \frac{(3+2^{22})\ell q\sigma}{2^n}. \qquad (36)$$

8.1 Proof of Theorem 10

This proof is almost same as the proof of Theorem 3. As in equation (9), we have

$$\mathbf{Adv}^{\pm\widetilde{\text{prp}}}_{\text{XCBv1[3Perm}(n)\text{]}}(A) \leq \mathbf{Adv}^{\pm\text{rnd}}_{\text{XCBv1[3Perm}(n)\text{]}}(A) + \binom{q}{2}\frac{1}{2^n}. \tag{37}$$

To bound $\mathbf{Adv}^{\pm\text{rnd}}_{\text{XCBv1[3Perm}(n)\text{]}}(A)$ we use the same sequence of games as we did for the proof of Theorem 3. Finally, we arrive at Game RND2, which is shown in Figure 4. And, we have,

$$\begin{aligned}\mathbf{Adv}^{\pm\text{rnd}}_{\text{XCBv1[3Perm}(n)\text{]}}(A) &\leq \Pr[A^{\text{RND2}} \text{ sets } \mathsf{bad}] \\ &\leq \sum_{1\leq i\leq 3} \Pr[\mathsf{COLR}_i] + \Pr[\mathsf{COLD}_i]. \end{aligned}\tag{38}$$

Where COLD_i and COLR_i denote the events of a collision in Dom_i and Rng_i respectively. The sets Dom_i and Rng_i are specified in the game RND2 in Figure 4. Now the main task is to bound the collision probabilities in the sets Dom_i and Rng_i.

After q queries of A, according to the game RND2, the elements in the sets Dom_1, Dom_2, Dom_3 and Rng_1, Rng_2, Rng_3 would be

$\mathsf{Dom}_1 = \{P_1^s : 1 \leq s \leq q\}$,
$\mathsf{Dom}_2 = \bigcup_{s=1}^{q}\{\mathsf{inc}^j(S^s) : 0 \leq j \leq m^s - 2\}$,
$\mathsf{Dom}_3 = \{C_1^s : 1 \leq s \leq q\}$.

$\mathsf{Rng}_1 = \{CC^s : 1 \leq s \leq q\}$,
$\mathsf{Rng}_2 = \bigcup_{s=1}^{q}\{Y_j^s : 0 \leq j \leq m^s - 2\}$,
$\mathsf{Rng}_3 = \{MM^s : 1 \leq s \leq q\}$.

As before, we consider the s-th query to consist of m^s blocks of the message and t^s blocks of the tweak. It may be the case that the last block of message and tweak has less than n bits. Moreover, in the ensuing analysis, the definition of $\ell^{s,s'}$ is changed to $\ell^{s,s'} = \max\{m^s + t^s, m^{s'} + t'\}$.

We have the following bounds on the collision probabilities.

Claim 11. *1.* $\Pr[\mathsf{COLD}_1] \leq \binom{q}{2}/2^n$.
2. $\Pr[\mathsf{COLD}_2] \leq \frac{\ell q \sigma \alpha_{max}}{2^n}$
3. $\Pr[\mathsf{COLD}_3] \leq \binom{q}{2}/2^n$.
4. $\Pr[\mathsf{COLR}_1] \leq \frac{(q-1)\sigma}{2^n}$
5. $\Pr[\mathsf{COLR}_2] \leq \binom{\sum_s m_s - q}{2}\frac{1}{2^n}$
6. $\Pr[\mathsf{COLR}_3] \leq \frac{(q-1)\sigma}{2^n}$

Proof. The proof of (1) and (3) is same as the proof of Claim 4, and proof of (5) is same as that of Claim 8.
Proof of (2): As in the proof of Claim 5, we have $\mathsf{Dom}_2 = D_1 \cup D_2 \cup \cdots \cup D_q$, where

$$D_s = \{\mathsf{inc}^j(S^s) : 0 \leq j \leq m^s - 2\}, \text{ for } 1 \leq s \leq q,$$

and

$$S^s = \begin{cases} CC^s \oplus H_{h_1}(P_2^s||P_3^s||\cdots||P_{m^s}^s, T^s) & \text{if } ty^s = enc \\ MM^s \oplus H_{h_2}(C_2^s||C_3^s||\cdots||C_{m^s}^s, T^s) & \text{if } ty^s = dec \end{cases}$$

Let us denote $P_2^s||P_3^s||\cdots||P_{m^s}^s$ by Q^s and $C_2^s||C_3^s||\cdots||C_{m^s}^s$ by R^s. Now, we have the following cases:

Respond to the s^{th} adversary query as follows:

ENCIPHER query $\mathbf{Enc}(T^s; P^s)$

10. $(P_1^s, P_2^s, \ldots, P_{m^s-1}^s, P_{m^s}^s) \leftarrow \mathsf{parse}_n(P^s)$
11. $ty^s = \mathsf{Enc}$
12. $C_1^s||C_2^s||\ldots||C_{m^s-1}^s||D_{m^s}^s \xleftarrow{\$} \{0,1\}^{nm^s}$
13. $C_{m^s}^s \leftarrow \mathsf{drop}_{n-r^s}(D_{m^s})$
14. **return** $C_1^s||C_2^s||\ldots||C_{m^s}^s$

DECIPHER query $\mathsf{Dec}(T^s; C^s)$

20. $(C_1^s, C_2^s, \ldots, C_{m^s-1}^s, C_{m^s}^s) \leftarrow \mathsf{parse}_n(C^s)$
21. $ty^s = \mathsf{Dec}$
22. $P_1^s||P_2^s||\ldots||P_{m^s-1}^s||V_{m^s}^s \xleftarrow{\$} \{0,1\}^{nm^s}$
23. $P_{m^s}^s \leftarrow \mathsf{drop}_{n-r^s}(V_{m^s})$
24. **return** $P_1^s||P_2^s||\ldots||P_{m^s}^s$

Finalization:

001. $h_1 \xleftarrow{\$} \{0,1\}^n$
002. $h_2 \xleftarrow{\$} \{0,1\}^n$
for $s = 1$ to q,
if $ty^s = \mathsf{Enc}$ **then**
101. **if** $P_1^s = P_1^{s'}$ for $s' < s$ **then**
102. $CC^s \leftarrow CC^{s'}$
103. **else**
104. $CC^s \xleftarrow{\$} \{0,1\}^n$
105. $\mathsf{Dom}_1 \leftarrow \mathsf{Dom}_1 \cup \{P_1^s\}$
106. $\mathsf{Rng}_1 \leftarrow \mathsf{Rng}_1 \cup \{CC^s\}$
107. **end if**
108. $S^s \leftarrow CC^s \oplus H_{h_1}(P_2^s||\ldots||P_{m^s}^s, T^s)$
109. $MM^s \leftarrow S^s \oplus H_{h_2}(C_2^s||\ldots||C_{m^s}^s, T^s)$
110. $\mathsf{Dom}_3 \leftarrow \mathsf{Dom}_3 \cup \{C_1^s\}$
111. $\mathsf{Rng}_3 \leftarrow \mathsf{Rng}_3 \cup \{MM^s\}$
112. **for** $i = 0$ to $m^s - 3$,
113. $Y_i^s \leftarrow C_{i+2}^s \oplus P_{i+2}^s$
114. $\mathsf{Dom}_2 \leftarrow \mathsf{Dom}_2 \cup \{\mathsf{inc}^i(S^s)\}$
115. $\mathsf{Rng}_2 \leftarrow \mathsf{Rng}_2 \cup \{Y_i^s\}$
116. **end for**
117. $Y_{m^s-2}^s \leftarrow \mathsf{pad}(P_{m^s}^s) \oplus D_{m^s}^s$
118. $\mathsf{Dom}_2 \leftarrow \mathsf{Dom}_2 \cup \{\mathsf{inc}^{m^s-2}(S^s)\}$
119. $\mathsf{Rng}_2 \leftarrow \mathsf{Rng}_2 \cup \{Y_{m^s-2}^s\}$
else if $ty^s = \mathsf{Dec}$ **then**
201. **if** $C_1^s = C_1^{s'}$ for $s' < s$ **then**
202. $MM^s \leftarrow MM^{s'}$
203. **else**
204. $MM^s \xleftarrow{\$} \{0,1\}^n$
205. $\mathsf{Dom}_3 \leftarrow \mathsf{Dom}_3 \cup \{C_1^s\}$
206. $\mathsf{Rng}_3 \leftarrow \mathsf{Rng}_3 \cup \{MM^s\}$
207. **end if**
208. $S^s \leftarrow MM^s \oplus H_{h_2}(C_2^s||\ldots||C_{m^s}^s, T^s)$
209. $CC^s \leftarrow S^s \oplus H_{h_1}(P_2^s||\ldots||P_{m^s}^s, T^s)$
210. $\mathsf{Dom}_1 \leftarrow \mathsf{Dom}_1 \cup \{P_1^s\}$
211. $\mathsf{Rng}_1 \leftarrow \mathsf{Rng}_1 \cup \{CC^s\}$
212. **for** $i = 0$ to $m^s - 3$,
213. $Y_i^s \leftarrow C_{i+2}^s \oplus P_{i+2}^s$
214. $\mathsf{Dom}_2 \leftarrow \mathsf{Dom}_2 \cup \{\mathsf{inc}^i(S^s)\}$
215. $\mathsf{Rng}_2 \leftarrow \mathsf{Rng}_2 \cup \{Y_i^s\}$
216. **end for**
217. $Y_{m^s-2}^s \leftarrow \mathsf{pad}(C_{m^s}^s) \oplus V_{m^s}^s$
218. $\mathsf{Dom}_2 \leftarrow \mathsf{Dom}_2 \cup \{\mathsf{inc}^{m^s-2}(S^s)\}$
219. $\mathsf{Rng}_2 \leftarrow \mathsf{Rng}_2 \cup \{Y_{m^s-2}^s\}$
end if
end for

SECOND PHASE
$\mathsf{bad} = \mathsf{false}$;
if (some value occurs more than once in $\mathsf{Dom}_i, i = 1, 2, 3$) **then bad** = true **end if**;
if (some value occurs more than once in $\mathsf{Rng}_i, i = 1, 2, 3$) **then bad** = true **end if**.

Fig. 4. Game RND2 for XCBv1.

Case 1: $ty^s = ty^{s'} = enc$

Sub case 1a: $(Q^s, T^s) = (Q^{s'}, T^{s'})$. In this case $P_1^s \neq P_1^{s'}$, which means that CC^s and $CC^{s'}$ are uniformly and independently distributed in $\{0,1\}^n$, which implies $\Pr[\mathsf{inc}^i(S^s) \oplus S^{s'}] \leq 1/2^n$.

Sub case 1b: $(Q^s, T^s) \neq (Q^{s'}, T^{s'})$. By Lemma 1, we have $\Pr[\mathsf{inc}^i(S^s) \oplus S^{s'}] \leq \alpha_{max}\ell^{s,s'}/2^n$

Case 2: $ty^s = ty^{s'} = dec$. It has the same subcases as in the previous case, and the bounds on the probability on the subcases are same as in the cases 1a and 1b.

Case 3: $ty^s = enc$ and $ty^{s'} = dec$. By Lemma 1 we have $\Pr[\mathsf{inc}^i(S^s) \oplus S^{s'}] \leq \alpha_{max}\ell^{s,s'}/2^n$

Following the same arguments as in the proof of Claim 5, we get

$$\begin{aligned}\Pr[\mathsf{COLD}_2] &\leq \sum_{1\leq s<s'\leq q} \frac{\alpha_{max}(m^s + m^{s'} - 4)\ell^{s,s'}}{2^n} \\ &\leq \frac{\ell q \sigma \alpha_{max}}{2^n} \end{aligned} \tag{39}$$

Proof of (4) If $ty^s = enc$, then CC^s is selected uniformly at random from $\{0,1\}^n$, and if $ty^s = ty^{s'} = enc$ and $P_1^s = P_1^{s'}$, then CC^s does not enter Rng_1. Thus if ty^s or $ty^{s'}$ is enc, then $\Pr[CC^s = CC^{s'}] = 1/2^n$. Next we settle the case $ty^s = ty^{s'} = dec$.

If $ty^s = dec$, then

$$CC^s = MM^s \oplus H_{h_2}(R^s, T^s) \oplus H_{h_1}(Q^s, T^s),$$

where $Q^s = P_2^s||P_3^s||\cdots||P_{m^s}^s$ and $R^s = C_2^s||C_3^s||\cdots||C_{m^s}^s$. We have the following cases to settle.

1. $(R^s, T^s) = (R^{s'}, T^{s'})$. In this case MM^s and $MM^{s'}$ are selected uniformly and independently from $\{0,1\}^n$, hence $\Pr[CC^s = CC^{s'}] \leq 1/2^n$.
2. $(R^s, T^s) \neq (R^{s'}, T^{s'})$. In this case we have

$$CC^s \oplus CC^{s'} = MM^s \oplus MM^{s'} \oplus \left[H_{h_2}(R^s, T^s) \oplus H_{h_2}(R^{s'}, T^{s'})\right] \oplus \left[H_{h_1}(Q^s, T^s) \oplus H_{h_1}(Q^{s'}, T^{s'})\right].$$

Let

$$\begin{aligned}\mu^{s,s'} &= MM^s \oplus MM^{s'}, \\ H_2^{s,s'} &= H_{h_2}(R^s, T^s) \oplus H_{h_2}(R^{s'}, T^{s'}), \\ H_1^{s,s'} &= H_{h_1}(Q^s, T^s) \oplus H_{h_1}(R^{s'}, T^{s'}).\end{aligned}$$

Note that $H_1^{s,s'} \oplus H_2^{s,s'}$ is a non zero bivariate polynomial on h_1, h_2 with (total) degree $\ell^{s,s'}$. Hence by Schwartz-Zippel Theorem

$$\Pr[CC^s \oplus CC^{s'} = 0] \leq \frac{\ell^{s,s'}}{2^n}$$

Hence we have

$$\begin{aligned}\Pr[\mathsf{COLR}_1] &\leq \sum_{1\leq s<s'\leq q} \frac{\ell^{s,s'}}{2^n} \\ &\leq \frac{(q-1)\sigma}{2^n}.\end{aligned}$$

Proof of (5): From the game RND2, we have for $1 \leq s \leq q$, $Y_j^s = C_{j+2}^s \oplus P_{j+2}^s$, where $0 \leq j \leq m^s - 3$, and

$$Y_{m^s-2}^s = \begin{cases} \mathsf{pad}(P_{m_s}^s) \oplus D_{m_s}^s & \text{if } s = enc \\ \mathsf{pad}(C_{m_s}^s) \oplus V_{m_s}^s & \text{if } s = dec. \end{cases}$$

Hence, there are $\sum_s m_s - q$ uniformly and independently generated n-bit strings in Rng_2. Hence the claim follows.

Proof of (6) is same as proof of (4). □

Now, using Claim 11 and Eqs. (37) and (38) we obtain the claimed bound.

9 The Bound with Practical Parameter Values

In this section, we compare the bounds that we derived with the bounds for other TES considering practical values of the parameters. We assume 2^{32} bytes of ciphertext/plaintext is available to the adversary in a hard disk of sector size 4096 bytes. We assume an underlying block cipher with a block length of 16 bytes. Thus, each message is of length 4096 bytes, i.e., 2^8 blocks. The total cipher/plaintext available to the adversary is $2^{32}/2^4 = 2^{28}$ blocks. And the total cipher/plaintext comprises of $2^{28}/2^8 = 2^{20}$ messages, which means 2^{20} queries and tweaks. Thus, the total query complexity of the adversary is $2^{28} + 2^{20}$. These parameter values are summarized in Table 4.

The claimed bounds and their numerical values based on the parameters stated above are summarized in Table 5. Unlike the other TES, the security bounds of both XCBv1 and XCBv2fb depends on the maximum allowed message length. The bounds stated in Theorems 3 and 10 assumes that the maximum message length is 2^{32} blocks as mentioned by the designers. The 2^{22} term in the bound comes from the maximum value of α_r where $1 \leq r \leq 2^{32}$. If the allowed length of messages is smaller, then both XCBv1 and XCBv2fb would have better bounds. If all messages are at most 2^8 blocks long, then 2^{22} can be replaced by $a = \max_{0 \leq r \leq 2^8} \alpha_r$. By computation, we found that $a = 2^{11}$. This can also be seen in Table 6 of [9], where a table of values of α_r for different r is reported. The modified bounds and its numerical values for XCBv1 and XCBv2fb based on the selected parameters is shown in Table 5.

Block length	n	128	We assume AES
Max query length	ℓ	$\frac{4\times 2^{10}}{2^4} + 1 = 2^8 + 1$	Assuming 4096-byte messages
Numb. of Queries	q	$\frac{2^{28}}{2^8} = 2^{20}$	Assuming 2^{32} bytes of cipher/plaintext is available to the adversary. Each message/cipher is 4096 bytes.
Query Complexity	σ	$\frac{2^{32}}{2^4} + \frac{2^{28}}{2^8} = 2^{28.006}$	Blks. of cipher/plaintext + tweak

Table 4. The values of the parameters

Modes	Source	Claimed bound	Numerical Value
CMC	[7]	$\frac{7\sigma^2}{2^n}$	$2^{-69.18}$
EME	[8]	$\frac{7\sigma^2}{2^n}$	$2^{-69.18}$
HCTR	[2]	$\frac{4.5\sigma^2}{2^n}$	$2^{-69.81}$
TET	[6]	$\frac{3\sigma^2}{2\phi(2^n-1)}$	$2^{-69.40}$
HEH, HMCH	[15]	$\frac{20\sigma^2}{2^n}$	$2^{-67.66}$
XCB (claimed)	[12]	$\frac{8q^2(\ell+2)^2}{2^n}$	$2^{-68.96}$
XCBv2fb	This paper	$\left(\frac{(5+2^{22})\ell q\sigma}{2^n}, \frac{(5+2^{11})\ell q\sigma}{2^n}\right)$	$(2^{-49.98}, 2^{-60.98})$
XCBv1	This paper	$\left(\frac{(3+2^{22})\ell q\sigma}{2^n}, \frac{(3+2^{11})\ell q\sigma}{2^n}\right)$	$(2^{-49.98}, 2^{-60.98})$

Table 5. Comparison of the bounds: q, σ, ℓ are the number of queries, query complexity, and number of blocks in the longest query respectively. Here ϕ is the Euler's totient. The two bounds for the two versions of XCB from this paper correspond to maximum message lengths of 2^{32} and 2^8 blocks. The numerical values are computed using the parameters in Table 4.

10 Conclusion

In this paper we took a close look at XCB. Based on the study we can conclude the following:

1. XCBv2 as specified in [12] is not secure as a TES. We found an easy distinguishing attack on XCBv2. The attack works because of a faulty padding scheme, and there seems to be no easy way to fix this problem. However, if the inputs to XCBv2 are such that their lengths are multiples of the block length of the block cipher, then our attack does not work. For this restricted message space XCBv2fb (the full block version of XCBv2) is secure.
2. Even for the restricted message space, XCBv2fb (possibly) does not have the security bound as claimed in [12]. This is due to the fact that the proof of the security theorem in [12] is wrong. The error stems from a faulty calculation of collision probabilities in the inc function. We point out the mistake by showing concrete examples where that the bound on the collision probabilities in the inc function as given in [12] are violated. These examples are highly motivated by a prior study in [9].
3. We provide a corrected security analysis for XCBv2fb which is supported by a detailed proof. The correct security bound that can be proved for XCBv2fb is worse than that claimed in [12].
4. XCBv1 does not suffer from the weaknesses as in XCBv2. The distinguishing attack which we present for XCBv2 does not work for XCBv1. XCBv1 (as specified in [11]) is a secure TES. There was no proof of the fact that XCBv1 is secure. We provide the first proof of security for XCBv1 along with a concrete security bound.
5. XCBv2 was derived as a small modification of XCBv1. The authors said that the modifications were made to enable easy analysis [12]. Though it is not very clear to us, how these modifications help in the analysis. Our analysis reveals that any modification in an existing cryptographic scheme should be done with utmost care, even an innocent looking change may have a grave impact on the security of the scheme.
6. XCBv2 is a part of the standard IEEE Std 1619.2-2010. Our analysis puts into serious doubts the methodology adopted by the working group for formulating the standard. We are surprised that an international standardization committee for a cryptographic scheme overlooked some important security issues, which were not so difficult to detect. Thus, our analysis of XCB indicates that contrary to the popular convention of blindly adopting standards, the outcomes of standardization efforts should also be critically analyzed before deploying them in a real application.

References

1. IEEE Std 1619.2-2010: IEEE standard for wide-block encryption for shared storage media. IEEE Computer Society, March 2011. `http://standards.ieee.org/findstds/standard/1619.2-2010.html`.
2. D. Chakraborty and M. Nandi. An improved security bound for HCTR. In *FSE*, pages 441–455, 2008.
3. D. Chakraborty and P. Sarkar. A new mode of encryption providing a tweakable strong pseudo-random permutation. In *FSE*, pages 293–309, 2006.
4. Debrup Chakraborty and Palash Sarkar. HCH: A new tweakable enciphering scheme using the hash-counter-hash approach. *IEEE Transactions on Information Theory*, 54(4):1683–1699, 2008.
5. S. Halevi. EME*: Extending EME to handle arbitrary-length messages with associated data. In *INDOCRYPT*, pages 315–327, 2004.
6. S. Halevi. Invertible universal hashing and the tet encryption mode. In *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 412–429. Springer, 2007.
7. S. Halevi and P. Rogaway. A tweakable enciphering mode. In *CRYPTO*, pages 482–499, 2003.
8. S. Halevi and P. Rogaway. A parallelizable enciphering mode. In *CT-RSA*, pages 292–304, 2004.
9. T. Iwata, K. Ohashi, and K. Minematsu. Breaking and repairing GCM security proofs. In *Advances in Cryptology - Crypto 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 31–49. Springer, 2012.
10. Cuauhtemoc Mancillas-López, Debrup Chakraborty, and Francisco Rodríguez-Henríquez. Reconfigurable hardware implementations of tweakable enciphering schemes. *IEEE Trans. Computers*, 59(11):1547–1561, 2010.

11. D. A. McGrew and S. R. Fluhrer. The extended codebook (XCB) mode of operation. Cryptology ePrint Archive, Report 2004/278, 2004.
12. D. A. McGrew and S. R. Fluhrer. The security of the extended codebook (XCB) mode of operation. In Carlisle Adams, Ali Miri, and Michael Wiener, editors, *Selected Areas in Cryptography*, volume 4876 of *Lecture Notes in Computer Science*, pages 311–327. Springer Berlin Heidelberg, 2007.
13. D. A. McGrew and J. Viega. Arbitrary block length mode, 2004. `http://grouper.ieee.org/groups/1619/email/pdf00005.pdf`.
14. P. Sarkar. Improving upon the TET mode of operation. In *ICISC*, volume 4817 of *Lecture Notes in Computer Science*, pages 180–192. Springer, 2007.
15. P. Sarkar. Efficient tweakable enciphering schemes from (block-wise) universal hash functions. *Information Theory, IEEE Transactions on*, 55(10):4749–4760, 2009.
16. P. Wang, D. Feng, and W. Wu. HCTR: A variable-input-length enciphering mode. In *CISC*, pages 175–188, 2005.

A XCB in IEEE 1619.2

Here we describe XCB verbatim as described in IEEE-std 1619.2, 2010.

1. $H \leftarrow \text{AES-Enc}(K, 0^{128})$
2. $K_e \leftarrow \text{msb}_k(\text{AES-Enc}(K, 0^{125}|001_2)|\text{AES-Enc}(K, 0^{125}|010_2))$
3. $K_d \leftarrow \text{msb}_k(\text{AES-Enc}(K, 0^{125}|011_2)|\text{AES-Enc}(K, 0^{125}|100_2))$
4. $K_c \leftarrow \text{msb}_k(\text{AES-Enc}(K, 0^{125}|101_2)|\text{AES-Enc}(K, 0^{125}|110_2))$
5. $A \leftarrow P[m-128 : m-1]$
6. $B \leftarrow P[0 : m-127]$
7. $C \leftarrow \text{AES-Enc}(K_e, A)$
8. $D \leftarrow C \oplus h_1(H, Z, B)$
9. $E \leftarrow B \oplus c(K_c, D, \#B)$
10. $F \leftarrow D \oplus h_2(H, Z, E)$
11. $G \leftarrow \text{AES-Dec}(K_d, F)$
12. $CT \leftarrow E|G$

In the above the length of the plaintext P is m bits. Note that the length of B is $m-126$ bits and this is also the length of E. The length of G is 128 bits and so the length of CT is $m-126+128 = m+2$ bits. So, applying the encryption function increases the length by 2 bits.