


# Small Stretch Problem of the DCT Scheme and How to Fix it

Yuchao Chen<sup>1,2</sup>, Tingting Guo<sup>3</sup>, Lei Hu<sup>4,5</sup>, Lina Shang<sup>6</sup>, Shuping Mao<sup>4,5</sup>, and Peng Wang<sup>7</sup>

<sup>1</sup> School of Cyber Science and Technology, Shandong University, Qingdao, China  
`chenyuchao@mail.sdu.edu.cn`

<sup>2</sup> Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, China

<sup>3</sup> Research Institute of Basic Theories, Zhejiang lab  
`guotingting4633@gmail.com`

<sup>4</sup> SKLOIS, Institute of Information Engineering, CAS  
{`hulei`, `maoshuping`}@`iie.ac.cn`

<sup>5</sup> School of Cyber Security, University of Chinese Academy of Sciences

<sup>6</sup> Space Star Technology Co., Ltd.  
`sln-8108@163.com`

<sup>7</sup> School of Cryptology, University of Chinese Academy of Sciences  
`w.rocking@gmail.com`

**Abstract.** DCT is a beyond-birthday-bound (BBB) deterministic authenticated encryption (DAE) mode proposed by Forler et al. in ACISP 2016, ensuring integrity by redundancy. The instantiation scheme of DCT employs the BRW polynomial, which is more efficient than the usual polynomial function in GCM by reducing half of the multiplication operations. However, we show that DCT suffers from a small stretch problem similar to GCM. When the stretch length  $\tau$  is small, choosing a special  $m$ -block message, we can reduce the number of queries required by a successful forgery to  $\mathcal{O}(2^\tau/m)$ . We emphasize that this attack efficiently balances space and time complexity, but does not contradict the security bounds of DCT. Finally, we propose an improved scheme named Robust DCT (RDCT) with a minor change to DCT, which improves the security when  $\tau$  is small and makes it resist the above attack.

**Keywords:** DCT · Deterministic Authenticated Encryption · AEAD · BRW Polynomial Function · Forgery Attack · Stretch.

## 1 Introduction

Authenticated encryption (AE) schemes [25] provide confidentiality and integrity simultaneously. AE achieves integrity by generating a tag or encoding some redundancy into the message, leading to ciphertext expansion, a.k.a. tag length or stretch. In the real world of cryptographic systems, such as RFID cards, sensor networks, or embedded devices, 128-bit tags may not be supported; instead, these embedded devices usually support tag sizes such as 32 bits or 64

bits. Generally, GCM [21] has a variety of tag lengths for choice by truncation, such as 64, 96, 104, 112, 120, and 128 bits, which fits all kinds of requirements. GCM is widely used in many applications, such as IPsec and TLS. NIST already standardizes GCM as SP 800-38D [7], and the ISO includes GCM as a part of ISO/IEC 19772:2020 [16].

However, GCM has a tag truncation problem due to the linear modification technique proposed by Ferguson [9] in 2005. When using a small truncated tag, adversaries can change the ciphertext by solving a system of linear equations (or a linear system) to obtain potential successful modifications with higher probability. For example, when GCM uses a 32-bit tag and adversaries know the ciphertext for a message consisting of  $2^{17}$  blocks (about 2 MB), with Ferguson’s technique, the probability of an adversary forging a 32-bit tag is  $2^{-16}$  instead of optimal  $2^{-32}$ . Furthermore, a successful forgery can also compromise the authentication key and lead to confidentiality attacks.

Many nonce-based AE schemes suffer catastrophic confidentiality and integrity failures when the nonce is repeated, including GCM [21] and OCB [26]. Hence, Rogaway and Shrimpton introduced the notion of Deterministic Authenticated Encryption (DAE) [27] at EUROCRYPT 2006, with the primary purpose of addressing the key-wrap issue, as well as the nonce-misuse issue.

Numerous DAE schemes have been proposed, including SIV [27], GCM-SIV [13], AES-GCM-SIV [12], and Deoxys-II [19]. All these schemes combine a conventional IV-based encryption scheme (e.g., CTR mode) and a PRF-secure authentication scheme. The security of SIV does not depend on the freshness of nonce. Since SIV requires two independent keys, it increases the key management overhead. In 2009, Iwata and Yasuda proposed a single-key mode of operation named HBS [18] to achieve the DAE goal. HBS also accelerated the speed by employing a polynomial universal hashing rather than blockcipher-based MAC. Subsequently, they proposed BTM [17], which requires only one blockcipher key as HBS and does not require the decryption algorithm of the underlying blockcipher, whereas HBS does.

Schemes like Deoxys-II [19] and HBS [18] suffer from the so-called birthday attack. Assuming that the block size of the underlying primitive is 128 bits, after about  $2^{64}$  queries, adversaries will obtain a successful attack with high probability. For example, Ferguson [8] proposed a birthday-bound forgery attack on OCB. BBB schemes are secure for above  $2^{n/2}$  queries, where  $n$  is the block size of the underlying primitive. BBB security is, therefore, a desirable goal for DAE.

In 2016, Forler et al. [10] proposed a beyond-birthday-bound DAE scheme named DCT (Deterministic Counter in Tweak), which is inspired by the CTRT (*CounTeR in Tweak*) encryption scheme [23] and the BRW polynomial [4]. DCT encodes  $\tau$  bits of redundancy and then encrypts it using the Hash-Counter [6, 22] approach to obtain a BBB DAE scheme. DCT can obtain different integrity strengths by selecting different values of  $\tau$ . They also proposed an efficient implementation that requires only a single key.

DCT uses a BBB encryption scheme, a  $2n$ -bit SPRP, and a BRW (Bernstein Rabin Winograd) polynomial to instantiate its underlying universal hash functions (UHF), which were proposed by Bernstein [4] in 2007, based on the work of Rabin and Winograd [24]. The BRW polynomial is faster than the UHFs used by GCM because the former requires only half as many multiplications over finite fields. The BRW polynomial is widely used in many schemes, including tweakable enciphering schemes [29, 30], message authentication codes (MACs) schemes [5], universal hash function [11], authenticated encryption schemes [10], etc.

**Our contributions.** In this paper, we describe several forgery attacks on the DCT scheme. Our attack results are summarized in Table 1.

- 1) We show that although DCT employs the BRW polynomial function to instantiate its UHF, it still suffers from a small stretch problem similar to that of GCM. When  $\tau$  is small, we consider the case where the message length is fixed in Section 5.2. Adversaries can linearize the UHFs of DCT by querying for particular structured  $(2^{u+2} - 2)$ -block messages where  $2 \leq u \leq \tau$ , and then attack the integrity of DCT with  $2^{\tau-u}$  decryption queries. This attack can efficiently balance the space complexity  $\mathcal{O}(2^{u+2})$  and the time complexity  $\mathcal{O}(2^{\tau-u})$  for a user-selected parameter  $u$ .
- 2) We extend this attack while the message length is no longer limited to  $2^{u+2} - 2$ , making our attacks more general. We further find that  $2^{u+2} - 3$  is the minimum message length needed to perform our forgery attack mentioned above when  $u$  is fixed.
- 3) To solve the above small stretch problem, we propose a variant of DCT named Robust DCT (RDCT) without minimal modification, and we prove the DAE security of RDCT. When  $\tau$  is small, our proof shows that a successful forgery attack requires  $\mathcal{O}(2^\tau)$  decryption queries.

**Table 1.** Comparing the attack complexity among GCM, DCT, and RDCT schemes.  $n$  is the size of the message block.  $m$  is the maximum number of blocks of a query.  $q$  is the number of queries.  $\tau$  is the number of bits in the GCM tag or the redundancy of DCT and RDCT.  $u$  is a user-selected parameter,  $2 \leq u \leq \tau$ .

Scheme	Provable security	Query complexity		Query length	Ref.
		Encryption	Decryption		
GCM	$\mathcal{O}(\frac{q^2 m^2}{2^n} + \frac{qm}{2^\tau})$	1	$2^{\tau-u}$	$2^{u+1}$	[9]
DCT	$\mathcal{O}(\frac{q^2 m^2}{2^{2n}} + \frac{qm^2}{2^\tau})$	1	$2^{\tau-u}$	$2^{u+2} - 2$ $2^{u+2} - 3$	Sect. 5.2 Sect. 5.3
RDCT	$\mathcal{O}(\frac{q^2 m^2}{2^{2n}} + \frac{q}{2^{\tau-q}})$	0	$2^\tau$	1	Sect. 6

Table 1 compares the complexity of the attack among GCM, DCT, and RDCT. GCM's attack requires one encryption query and  $2^{\tau-u}$  decryption queries

with  $2^{u+1}$  length messages. However, our attack on DCT requires a longer message length than the attack on GCM, but the query complexity for attacking both schemes is at the same magnitude. The core of our attack is to calculate the difference  $D$ , which only needs to be done once for both schemes. We remark that we only attack the DCT scheme with a small stretch length. When users select  $2n$ -bit stretch, the complexity of our attack is impracticable.

**Organization of the paper.** The paper is structured as follows: after Section 3 reviews the linear modification technique and Section 4 introduces the DCT scheme, Section 5 discusses the small stretch problem of DCT, Section 6 presents our new scheme named RDCT. Section 7 gives a summary of this work.

## 2 Preliminaries

### 2.1 Notations

We use lowercase letters  $x, y$  for integers, uppercase letters  $X, Y$  for strings or functions, and curlicue uppercase letters  $\mathcal{X}, \mathcal{Y}$  for sets. Let  $X\|Y$  represent the concatenation of strings  $X$  and  $Y$ , and let  $X \oplus Y$  represent the result of their bitwise XOR. We denote  $\emptyset$  as the empty set,  $|X|$  for the length of  $X$ ,  $x \xleftarrow{\$} \mathcal{X}$  for the element  $x$  chosen uniformly at random from the set  $\mathcal{X}$ . We define  $\Pr[V]$  as the probability of event  $V$ . We define  $\text{Perm}(\mathcal{S})$  as the set of all permutations on  $\mathcal{S}$ ;  $\widetilde{\text{Perm}}(\mathcal{T}, \mathcal{S})$  as the set of all tweakable permutations on  $\mathcal{S}$  with nonempty tweak space  $\mathcal{T}$ . We define  $\mathbb{Z}^+$  as the set of positive integers. We define  $MSB_\tau$  as the most significant  $\tau$  bits and  $LSB_\tau$  as the least significant  $\tau$  bits. We define  $\overline{X}$  as the 128-bit coefficients vector of a polynomial over  $\mathbb{GF}(2)$  corresponding to an element  $X \in \mathbb{GF}(2^{128})$ .

We assume that an adversary  $\mathbf{A}$  can interact with several given oracles as black boxes that run in time at most  $t$  and make at most  $q$  queries of at most  $m$  blocks in total, and we denote by  $\mathbf{A}^{\mathcal{O}} \Rightarrow b$  where  $b = 0$  or  $1$  the output of  $\mathbf{A}$  after interacting with an oracle  $\mathcal{O}$ . We write

$$\mathbf{Adv}_E^{\text{P}}(\mathbf{A}) := |\Pr[\mathbf{A}^E \Rightarrow 1] - \Pr[\mathbf{A}^I \Rightarrow 1]|,$$

as the advantage of  $\mathbf{A}$  to distinguish between oracles  $E$  and  $I$ , where P is the security goal,  $E$  is the real cryptographic primitive, and  $I$  is the ideal primitive. We define  $\mathbf{Adv}_E^{\text{P}}(q, m, t) := \max_{\mathbf{A}} \left\{ \mathbf{Adv}_E^{\text{P}}(\mathbf{A}) \right\}$  as the maximum of  $\mathbf{Adv}_E^{\text{P}}(\mathbf{A})$  over all adversaries against the P-security of  $E$  that run in time at most  $t$  and make at most  $q$  queries of at most  $m$  blocks in total.

### 2.2 Definition of Universal Hash Functions

**Definition 1 (Universal Hash Functions).** Let  $\mathcal{X} \subseteq \{0, 1\}^*$ ,  $\mathcal{Y} \subseteq \{0, 1\}^n$ ,  $\mathcal{K}$  is a key space. Let  $H : \mathcal{K} \times \mathcal{X} \rightarrow \mathcal{Y}$  denote hash function.  $H$  is called  $\epsilon$ -almost-universal ( $\epsilon$ -AU), such that for all distinct elements  $X, X' \in \mathcal{X}$ , it holds that  $\Pr[K \xleftarrow{\$} \mathcal{K} : H_K(X) = H_K(X')] \leq \epsilon$ .

$H$  is called  $\epsilon$ -almost-XOR-universal ( $\epsilon$ -AXU), such that for all distinct elements  $X, X' \in \mathcal{X}$  and  $Y \in \mathcal{Y}$ , it holds  $\Pr[K \xleftarrow{\$} \mathcal{K} : H_K(X) \oplus H_K(X') = Y] \leq \epsilon$ .

### 2.3 Security of (Tweakable) Blockciphers

**Definition 2 ((Strong) PRP Advantage).** Fix integers  $n, k \geq 1$ . Let  $E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$  be a blockcipher and  $\mathbf{A}(\mathbf{A}')$  be an adversary with access to an oracle (two oracles). Let  $K \xleftarrow{\$} \{0, 1\}^k$  and  $\pi \xleftarrow{\$} \text{Perm}(\{0, 1\}^n)$ . Then, the advantages PRP and SPRP of  $\mathbf{A}$  with respect to  $E$  are defined as  $\text{Adv}_E^{\text{PRP}}(\mathbf{A}) := |\Pr[\mathbf{A}^{E_K} \Rightarrow 1] - \Pr[\mathbf{A}^\pi \Rightarrow 1]|$  and  $\text{Adv}_{E, E^{-1}}^{\text{SPRP}}(\mathbf{A}) := |\Pr[\mathbf{A}^{E_K, E_K^{-1}} \Rightarrow 1] - \Pr[\mathbf{A}^{\pi, \pi^{-1}} \Rightarrow 1]|$ , respectively.

**Definition 3 ((Strong) Tweakable PRP Advantage).** Fix integers  $n, k \geq 1$ . Let  $\mathcal{T}$  denote a nonempty set. Let  $\tilde{E} : \{0, 1\}^k \times \mathcal{T} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$  be a tweakable blockcipher and  $\mathbf{A}$  be an adversary with access to an oracle (two oracles). Let  $K \xleftarrow{\$} \{0, 1\}^k$  and  $\tilde{\pi} \xleftarrow{\$} \widetilde{\text{Perm}}(\mathcal{T}, \{0, 1\}^n)$ . Then, the advantages TPRP and STPRP of  $\mathbf{A}$  with respect to  $\tilde{E}$  are defined as  $\text{Adv}_{\tilde{E}}^{\text{TPRP}}(\mathbf{A}) := |\Pr[\mathbf{A}^{\tilde{E}_K} \Rightarrow 1] - \Pr[\mathbf{A}^{\tilde{\pi}} \Rightarrow 1]|$  and  $\text{Adv}_{\tilde{E}, \tilde{E}^{-1}}^{\text{STPRP}}(\mathbf{A}) := |\Pr[\mathbf{A}^{\tilde{E}_K, \tilde{E}_K^{-1}} \Rightarrow 1] - \Pr[\mathbf{A}^{\tilde{\pi}, \tilde{\pi}^{-1}} \Rightarrow 1]|$ , respectively.

### 2.4 Security of IV-Based Encryption Schemes

Define the nonempty IV space  $\mathcal{IV}$ , the nonempty key space  $\mathcal{K}$ , the message space  $\mathcal{M} \subseteq \{0, 1\}^*$  and the ciphertext space  $\mathcal{C} \subseteq \{0, 1\}^*$ . An IV-based encryption scheme [2] is a tuple  $\Pi = (\mathcal{E}, \mathcal{D})$  of encryption  $\mathcal{E} : \mathcal{K} \times \mathcal{IV} \times \mathcal{M} \rightarrow \mathcal{C}$  and decryption  $\mathcal{D} : \mathcal{K} \times \mathcal{IV} \times \mathcal{C} \rightarrow \mathcal{M}$  algorithms. For any inquiries  $M$ , encryption oracle samples uniformly at random  $IV \xleftarrow{\$} \mathcal{IV}$  and computes the ciphertext  $C \leftarrow \mathcal{E}_K^{\text{IV}}(M)$ . The real oracle  $\mathcal{E}_K$  outputs  $(IV \| C)$ , and the random oracle  $\mathcal{E}^{\$}$  outputs with a random string as long as  $|IV \| \mathcal{E}_K^{\text{IV}}(M)|$ .

**Definition 4 (ivE Advantage).**  $K \xleftarrow{\$} \mathcal{K}$ , let  $\Pi = (\mathcal{E}, \mathcal{D})$  be an IV-based encryption scheme. Let  $\mathbf{A}$  be an adversary with access to an oracle. Then, the advantage  $\text{ivE}$  of  $\mathbf{A}$  over  $\Pi$  is defined as  $\text{Adv}_{\Pi}^{\text{ivE}}(\mathbf{A}) := |\Pr[\mathbf{A}^{\mathcal{E}_K} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathcal{E}^{\$}} \Rightarrow 1]|$ .

## 3 Linear Modification Technique

Each element  $X \in \mathbb{GF}(2^{128})$  can be represented by a 127-degree polynomial  $\text{Poly}(X)$  over  $\mathbb{GF}(2)$ . We denote  $\bar{X} = \begin{pmatrix} x_1 \\ \vdots \\ x_{128} \end{pmatrix}$  as the column vector of the coefficients of  $\text{Poly}(X)$  over  $\mathbb{GF}(2)$  and  $\bar{X}^T$  as the transpose of  $\bar{X}$ .

### 3.1 Multiplication operation

The function  $F_C(X) = C \cdot X$  over  $\mathbb{GF}(2^{128})$  with irreducible polynomial  $p(x)$  for a constant field element  $C$  is a linear function. For each  $C$  there exists a  $128 \times 128$  matrix  $M_C$  over  $\mathbb{GF}(2)$  such that

$$\overline{C \cdot X} = M_C \overline{X} = (M_C^1 \ M_C^2 \ \cdots \ M_C^{128}) \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{128} \end{pmatrix}$$

for all  $X \in \mathbb{GF}(2^{128})$ , where  $M_C^i$  represents the  $i$ -th column of  $M_C$ ,  $1 \leq i \leq 128$ . When  $\overline{X}$  is valued with an orthonormal basis from  $(1, 0, \dots, 0)^T$  through  $(0, \dots, 0, 1)^T$ , we can obtain the following equations to calculate  $M_C$ :

$$\begin{cases} \overline{C \cdot x^{127}} = M_C^1 \\ \vdots \\ \overline{C \cdot x} = M_C^{127} \\ \overline{C \cdot 1} = M_C^{128}. \end{cases}$$

Therefore, each column vector of  $M_C$  is linear combinations of  $\overline{C}$ , and  $\overline{C \cdot x^i}$  can be seen as the  $\overline{C}$  left-shifted by  $i$  bits,  $0 \leq i \leq 127$ . When it occurs overflow, we need to modulo the coefficient of  $p(x)$ .

### 3.2 Square operation

Due to the fact that  $\mathbb{GF}(2^{128})$  is a field of characteristic of 2, which implies that  $(A + B)^2 = A^2 + B^2$  for any  $A, B \in \mathbb{GF}(2^{128})$ . Therefore, the function  $F_S(X) = X^2$  over  $\mathbb{GF}(2^{128})$  with irreducible polynomial  $p(x)$  is a linear function. Thus, there exists a fixed matrix  $M_S$  such that

$$\overline{X^2} = M_S \overline{X} = (M_S^1 \ M_S^2 \ \cdots \ M_S^{128}) \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{128} \end{pmatrix}$$

for all  $X \in \mathbb{GF}(2^{128})$ . When  $\overline{X}$  is valued with an orthonormal basis from  $(1, 0, \dots, 0)^T$  through  $(0, \dots, 0, 1)^T$ , we can obtain the following equations to calculate  $M_S$ :

$$\begin{cases} \overline{(x^{127})^2} = M_S^1 \\ \vdots \\ \overline{x^2} = M_S^{127} \\ \overline{1} = M_S^{128}. \end{cases}$$

Therefore, each column vector of  $M_S$  can be seen as the  $\overline{1}$  left-shifted by  $2i$  bits; when it occurs overflow, we need to modulo the coefficient of  $p(x)$ . Note that  $M_S$  does not depend on anything except the chosen field representation. Therefore,  $M_S$  is a public matrix.

### 3.3 Concrete attack

This section introduces Ferguson’s linear modification technique [9], which attacks the GCM’s integrity by changing the ciphertext  $C := C_1 \| C_2 \| \dots \| C_m$  without changing the tag. Since Ferguson’s attack does not use the associated data  $A$ , we will ignore it. Then the authentication function can be denoted as

$$T := R \oplus \sum_{i=1}^m C_i H^i,$$

where  $R = E_K(0)$  is the authentication key, and  $H = E_K(N \| 1)$  is a 128-bit key generated by encrypting the nonce with a blockcipher  $E_K$ ,  $C_1 = (|A|, |C|)$  is length information of the inputs. The truncated tag comprises the most significant  $\tau$  bits of the  $T$ , denoted by  $MSB_\tau(T)$ . The attacking goal is to find  $\Delta := \Delta_1 \| \Delta_2 \| \dots \| \Delta_m$  such that

$$MSB_\tau \left( \sum_{i=1}^m \Delta_i H^i \right) = 0^\tau.$$

So we can obtain the collision

$$MSB_\tau(T) = MSB_\tau \left( T \oplus \sum_{i=1}^m \Delta_i H^i \right)$$

for any two ciphertexts  $C$  and  $C \oplus \Delta$  of equal length, which means if we query GCM to obtain a ciphertext triple  $(N, C, T)$ , we can forge another variable ciphertext triple  $(N, C \oplus \Delta, T)$  successfully. The concrete steps of searching  $\Delta$  are as follows.

1. Adjust the search goal to  $\Delta = \Delta_{2^0} \| \Delta_{2^1} \| 0^n \| \Delta_{2^2} \| 0^{3n} \| \Delta_{2^3} \| \dots$  where  $|\Delta_i| = n$ . That is to say, search coefficients  $D_i := \Delta_{2^i} (1 \leq 2^i \leq m)$  such that

$$MSB_u \left( \sum_i D_i H^{2^i} \right) = 0^u,$$

where the parameter  $u \leq \tau$ . We focus on  $D_i$ s for the reason that only they can make  $\sum_i D_i H^{2^i}$  linearization. Since the first ciphertext block encodes the length information, we do not change it and let  $D_0 = 0^n$ .

2. Represent the linear function  $T_1 = \sum_i D_i H^{2^i}$  in terms of  $H$  by matrix over finite fields. Consider  $T_1$  as bit vector  $\overline{T_1}$  and  $H$  as  $\overline{H}$ :

$$\overline{T_1} = \sum_i M_{D_i} (M_S)^i \overline{H},$$

where the matrix  $M_{D_i}$  represents the operation corresponding to multiplication with  $D_i$ , and the elements in  $M_{D_i}$  are all linear combinations of bits of  $D_i$ .  $M_S$  is a fixed matrix that represents the square operation.

To force  $u$  bits of the  $\overline{T_1}$  to zero, we need to create  $128 \times u$  linear equations about  $D_i$  such that  $u$  row of  $\sum_i M_{D_i} (M_S)^i$  is completely zero. When the number of  $D_i$  is  $u + 1$ , which corresponds to  $128 \times (u + 1)$  free variables (or unknowns), the number of unknowns exceeds the number of equations, we can obtain nontrivial solutions of  $\Delta$ , and the size of the solution set of the linear system is at least  $2^{128}$ .

3. We continue to perform about  $2^{\tau-u}$  decryption queries in search of the remaining  $\tau - u$  bits tag corresponding to the modified ciphertext  $C \oplus \Delta$ , leading to a successful forgery  $(N, C \oplus \Delta, T)$ .

For example, when GCM uses a 32-bit tag, the attack consists of the following steps: First, assume that adversaries can obtain the ciphertext for a message of  $m = 2^{17}$  blocks (about 2 MB) by encryption queries, which corresponds to  $17 \times 128$  unknowns. Second, suppose that the number of unknowns is greater than the number of linear equations; adversaries can calculate nontrivial solutions of  $D_1, D_2, \dots, D_{17}$  that make the 16 rows of the  $\sum_i M_{D_i}(M_S)^i$  equal to zero by creating  $16 \times 128$  constraint equations about  $D_i, 1 \leq i \leq 17$ . Third, adversaries continue to perform about  $q = 2^{16}$  decryption queries in search of the remaining 16 bits tag corresponding to the modified ciphertext, leading to a successful forgery.

More generally, assuming the length of the truncated tag is  $\tau$ , adversaries know the ciphertext of a message of  $m = 2^{u+1}$  blocks and can successfully obtain a forge with  $q = 2^v$  queries, which  $u + v = \tau$ . This technique can efficiently balance the number of message blocks  $m$  selected by the adversary and the number of queries  $q$  needed for the forgery.

## 4 A DAE Scheme: DCT

In this section, we introduce the notion of DAE [27] and the DCT scheme [10].

### 4.1 DAE Scheme

DAE scheme [27] is a tuple  $\tilde{\Pi} = (\tilde{\mathcal{E}}, \tilde{\mathcal{D}})$  of deterministic algorithms  $\tilde{\mathcal{E}} : \mathcal{K} \times \mathcal{A} \times \mathcal{M} \rightarrow \mathcal{C}$  and  $\tilde{\mathcal{D}} : \mathcal{K} \times \mathcal{A} \times \mathcal{C} \rightarrow \mathcal{M} \cup \{\perp\}$  with key space  $\mathcal{K}$ , associated-data space  $\mathcal{A}$ , and message/ciphertext space  $\mathcal{M}, \mathcal{C} \subseteq \{0, 1\}^*$ . For each  $K \in \mathcal{K}, A \in \mathcal{A}, M \in \mathcal{M}$ ,  $\tilde{\mathcal{E}}_K$  maps  $(A, M)$  to an output  $C$  such that  $|C| = |M| + \tau$  for fixed stretch length  $\tau$ .  $\tilde{\mathcal{D}}_K(A, C)$  outputs the corresponding message  $M$  if  $C$  is valid and  $\perp$  otherwise, where  $\perp$  is a symbol of decryption failures. DAE scheme can remove the overhead of nonce by exploiting the existing entropy or redundancy in the inputs; However, if nonce needs to be input when using a DAE scheme, users only need to take part of  $A$  as the nonce.

DAE scheme achieves both confidentiality and integrity. Confidentiality means that adversaries cannot obtain any information about plaintext from the corresponding ciphertext except the length; Integrity means adversaries cannot forge a fresh (not previously generated by the sender) and valid (pass the decryption oracle's verification) message. We define  $\text{DET}_{\text{PRIV}}, \text{DET}_{\text{AUTH}}$  as the confidentiality and integrity of the DAE scheme, respectively. The formalized definitions are as follows.

**Definition 5 (Confidentiality and Integrity Advantages).** Let  $\tilde{\Pi} = (\tilde{\mathcal{E}}, \tilde{\mathcal{D}})$  be a DAE scheme and  $K \leftarrow \mathcal{K}$ . Let  $\mathbf{A}$  be adversaries;  $\mathbf{A}$  has access to one or



two oracles  $\mathcal{O}_1$  and  $\mathcal{O}_2$ .  $\mathbf{A}$  are not allowed to ask  $\mathcal{O}_2$  as input the result of querying  $\mathcal{O}_1$ , and vice versa.  $\mathbf{A}$  does not repeat a query. Then, the DETPRIV and DETAUTH advantages of  $\mathbf{A}$  with respect to  $\tilde{\Pi}$ , are defined as

$$\begin{aligned} \text{Adv}_{\tilde{\Pi}}^{\text{DETPRIV}}(\mathbf{A}) &:= \left| \Pr \left[ \mathbf{A}^{\tilde{\mathcal{E}}_K} \Rightarrow 1 \right] - \Pr \left[ \mathbf{A}^{\mathcal{E}^{\tilde{\mathcal{E}}}} \Rightarrow 1 \right] \right|, \\ \text{Adv}_{\tilde{\Pi}}^{\text{DETAUTH}}(\mathbf{A}) &:= \Pr \left[ \mathbf{A}^{\tilde{\mathcal{E}}_K, \tilde{\mathcal{D}}_K} \text{ forges} \right], \end{aligned}$$

where “forges” means that  $\mathbf{A}$  asks  $\tilde{\mathcal{D}}_K$  with  $(A, C)$  and returns anything other than  $\perp$  for at least one query among multiple decryption queries.

Rogaway and Shrimpton [27] introduced the “All-in-One” definition, which is equivalent to the two-part notion that requires deterministic confidentiality DETPRIV and deterministic authenticity DETAUTH. The relation between it and DETPRIV and DETAUTH is as follows.

**Theorem 1 (All-in-One DAE Advantage [10, 28]).** *Let  $\tilde{\Pi} = (\tilde{\mathcal{E}}, \tilde{\mathcal{D}})$  be a DAE scheme and  $K \leftarrow \mathcal{K}$ . Let  $\mathbf{A}$  be a DAE adversary on  $\tilde{\Pi}$  with access to two oracles  $\mathcal{O}_1$  and  $\mathcal{O}_2$ ,  $\mathbf{A}$  are not allowed to ask  $\mathcal{O}_2$  as input the result of querying  $\mathcal{O}_1$ . Then, the All-in-One DAE advantages of  $\mathbf{A}$  with respect to  $\tilde{\Pi}$  is defined as*

$$\text{Adv}_{\tilde{\Pi}}^{\text{DAE}}(\mathbf{A}) := \left| \Pr \left[ \mathbf{A}^{\tilde{\mathcal{E}}_K, \tilde{\mathcal{D}}_K} \Rightarrow 1 \right] - \Pr \left[ \mathbf{A}^{\mathcal{E}^{\tilde{\mathcal{E}}}, \perp} \Rightarrow 1 \right] \right|,$$

where  $\mathbf{A}$  runs in time at most  $t$  and asks  $q_e$  queries to its left oracles, and asks  $q_d$  queries to its right oracles,  $\mathbf{A}$  asks at most  $m$  blocks in total. Then, there exists a DETPRIV adversary  $\mathbf{A}_1$  and a DETAUTH adversary  $\mathbf{A}_2$  both on  $\tilde{\Pi}$ , such that

$$\text{Adv}_{\tilde{\Pi}}^{\text{DAE}}(\mathbf{A}) \leq \text{Adv}_{\tilde{\Pi}}^{\text{DETPRIV}}(\mathbf{A}_1) + \text{Adv}_{\tilde{\Pi}}^{\text{DETAUTH}}(\mathbf{A}_2),$$

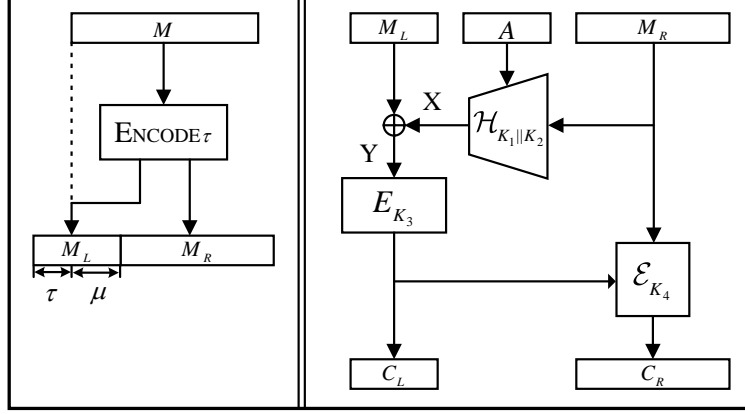
where  $\mathbf{A}_1$  and  $\mathbf{A}_2$  make at most  $q = q_e + q_d$  queries with a maximum of  $m$  blocks and run in time  $\mathcal{O}(t)$  each.

## 4.2 The DCT Scheme

Forler et al. [10] proposed the DCT scheme, a beyond-birthday-bound DAE scheme. We show the encryption of DCT in Figure 1.

Fix the parameters  $n, \tau \geq 1$  with  $\tau \leq 2n$ . Let  $\mu = 2n - \tau$ . Let  $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3$  and  $\mathcal{K}_4$  be nonempty key spaces and  $\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2 \times \mathcal{K}_3 \times \mathcal{K}_4$ . Let  $\mathcal{A} \subseteq \{0, 1\}^*$ ,  $\mathcal{M} \subseteq \{0, 1\}^{\geq \mu}$ ,  $\mathcal{C} \subseteq \{0, 1\}^{\geq 2n}$  denote the associated-data space, message space, and ciphertext space, respectively. Let  $\mathcal{H} : \mathcal{K}_1 \times \mathcal{K}_2 \times \mathcal{A} \times \{0, 1\}^* \rightarrow \{0, 1\}^{2n}$  be an AXU hash function with key space  $\mathcal{K}_1 \times \mathcal{K}_2$ . Let  $E : \mathcal{K}_3 \times \{0, 1\}^{2n} \rightarrow \{0, 1\}^{2n}$  be a permutation with key space  $\mathcal{K}_3$ . Let  $\Pi_1 = (\mathcal{E}, \mathcal{D})$  be an IV-based encryption scheme with key space  $\mathcal{K}_4$  and IV space  $\mathcal{IV} = \{0, 1\}^{2n}$ . Let  $\Pi_2 = (\text{ENCODE}_\tau, \text{DECODE}_\tau)$  be an encode scheme with encoding function and decoding function

$$\begin{aligned} \text{ENCODE}_\tau &: \mathcal{M} \rightarrow \{0, 1\}^{2n} \times \{0, 1\}^{|\mathcal{M}| - 2n + \tau}, \\ \text{DECODE}_\tau &: \{0, 1\}^{2n} \times \{0, 1\}^{|\mathcal{M}| - 2n + \tau} \rightarrow \mathcal{M} \cup \{\perp\}, \end{aligned}$$



**Fig. 1.** The  $\text{ENCODE}_\tau$  (left) process. The encryption process of DCT (right).

where  $\text{ENCODE}_\tau$  is an injection, encodes  $\tau$  bits redundancy into the input. The  $\tau$  bits redundancy is fully contained in the left part of the output. For example,  $\text{ENCODE}_\tau(M) = (M_L, M_R)$  where  $M_L = 0^\tau \| \text{MSB}_\mu(M)$  and  $M_R = \text{LSB}_{|M|-2n+\tau}(M)$ . The decoding function returns a unique  $M \in \mathcal{M}$  such that  $\text{ENCODE}_\tau(M) = (X, Y)$  for  $(X, Y) \in \{0, 1\}^{2n} \times \{0, 1\}^{|M|-2n+\tau}$  if such an  $M$  exists; otherwise, it returns  $\perp$ . Then the DCT scheme  $\text{DCT}_{\mathcal{H}, E, \Pi_1, \Pi_2} = (\tilde{\mathcal{E}}, \tilde{\mathcal{D}})$  based on  $\mathcal{H}, E, \Pi_1$  and  $\Pi_2$  is in Algorithm 1.

---

**Algorithm 1** Encryption and decryption of the DCT scheme

---

- |                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>1: <b>function</b> <math>\tilde{\mathcal{E}}_{K_1, K_2, K_3, K_4}(A, M)</math></p> <p>2:   <math>(M_L, M_R) \leftarrow \text{ENCODE}_\tau(M)</math></p> <p>3:   <math>C_L \leftarrow E_{K_3}(M_L \oplus \mathcal{H}_{K_1 \  K_2}(A, M_R))</math></p> <p>4:   <math>C_R \leftarrow \mathcal{E}_{K_4}(C_L, M_R)</math></p> <p>5:   <b>return</b> <math>(C_L \  C_R)</math></p> <p>6: <b>end function</b></p> | <p>7: <b>function</b> <math>\tilde{\mathcal{D}}_{K_1, K_2, K_3, K_4}(A, C)</math></p> <p>8:   <math>(C_L, C_R) \leftarrow C</math></p> <p>9:   <math>M_R \leftarrow \mathcal{D}_{K_4}(C_L, C_R)</math></p> <p>10:   <math>M_L \leftarrow E_{K_3}^{-1}(C_L) \oplus \mathcal{H}_{K_1 \  K_2}(A, M_R)</math></p> <p>11:   <b>return</b> <math>\text{DECODE}_\tau(M_L \  M_R)</math></p> <p>12: <b>end function</b></p> |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
- 

### 4.3 Instantiation of DCT

The instantiation scheme of DCT employs a CTR-like encryption scheme as  $\mathcal{E}_{K_4}(IV, M) = e_{K_4}(IV) \oplus M$  where  $e_{K_4}$  generates a  $|M|$ -bit stream. Due to the nonce length of the  $\Pi_1$  being  $2n$  bits, DCT employs a  $2n$ -bit permutation as  $E$ .  $\Pi_2$  encodes the  $\tau$  bits zero into the message. DCT uses the BRW polynomial to instantiate its underlying UHFs.

The attack in Section 5 focuses on the authentication security of the DCT instantiation scheme. In the next section, we will show that DCT suffers from the so-called small stretch problem.

## 5 Attacks to DCT with Small Stretch

In Section 5.1, we give a brief introduction to the UHF's used by DCT and the detailed steps of our attack. In Section 5.2, we first analyze the small stretch problem of DCT with a special fixed message length. Next, in Section 5.3, we consider the case while message length is no longer limited as a fixed value. Finally, in Section 5.4, we give a brief analysis of the security bounds of DCT.

### 5.1 The Universal Hash Function in DCT

In DCT, the universal hash function based on the BRW polynomial can be denoted as:

$$\mathcal{H}_{K_1\|K_2}(M) = KBRW_{K_1}(M_1, \dots, M_m) \| KBRW_{K_2}(M_1, \dots, M_m),$$

where  $K_1, K_2 \in \{0, 1\}^n$  are independent keys and  $KBRW_K(M) = K \cdot BRW_K(M)$  is defined directly as follows.

**Definition 6 (KBRW Polynomial Function [10]).** *Given a  $m$ -block message  $M = (M_1, \dots, M_m)$ ,  $K \in \{0, 1\}^n$ , the polynomial function  $KBRW_K(M)$  is defined as follows:*

$$\begin{aligned} KBRW_K(\varepsilon) &= 0^n; \\ KBRW_K(M_1) &= M_1K; \\ KBRW_K(M_1, M_2) &= M_1K^2 \oplus M_2K; \\ KBRW_K(M_1, M_2, M_3) &= K^4 \oplus M_1K^3 \oplus M_2K^2 \oplus (M_1M_2 \oplus M_3)K; \\ KBRW_K(M_1, \dots, M_m) &= KBRW_K(M_1, \dots, M_{t-1})(K^t \oplus M_t) \oplus \\ &KBRW_K(M_{t+1}, \dots, M_m) \text{ if } t \leq m < 2t \text{ for } t = 2^i, i \geq 2. \end{aligned}$$

where  $\varepsilon$  represents the empty string. All operations in the KBRW function are performed over  $\mathbb{GF}(2^n)$ , the Galois Field with a given irreducible polynomial  $p(x)$  of degree  $n$ . For  $n = 128$ ,  $p(x) = x^{128} + x^7 + x^2 + x + 1$ .

Since the associated data  $A$  are irrelevant to our attacks, we ignore them for convenience. For DCT with  $\tau \leq 2n$  and  $A = \varepsilon$ , when we obtain the ciphertext  $C_L \| C_R$  corresponding to  $(A, M) = (\varepsilon, M_L \| M_R)$ , we have the following equation:

$$MSB_\tau(M_L \oplus KBRW_K(M_R)) = 0^\tau.$$

We can query the decryption of DCT with  $(A, C_L \| C'_R)$  where only the value of  $C_R$  is modified to  $C'_R$ . Note that the CTR-like encryption scheme, if and only if the following equation establishes that the forgery is successful:

$$MSB_\tau(M_L \oplus KBRW_K(M_R \oplus C_R \oplus C'_R)) = 0^\tau.$$

So the forgery attack is reduced to the problem of looking for a modification string  $D = C_R \oplus C'_R = M_R \oplus M'_R$  while keeping  $MSB_\tau(KBRW_K(M)) = MSB_\tau(KBRW_K(M \oplus D))$ , the same problem in Section 3 but with a different universal hash function.

The core of our attack is to choose a particular structured message  $M$  and set some blocks of  $D$  as unknowns, making  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$  a linear function of  $K$ . The generic steps of the attack are as follows:

1. By selecting a particular structured message  $M = M_L \| M_R$  to query the encryption of DCT, we can obtain the corresponding ciphertext  $(C_L, C_R)$ .
2. Explore using the technique stated in Theorem 2, we can make  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$  become a linear function of  $K$ ; After that, using the technique outlined in Section 3 for this linearized function, we can calculate a set of solutions  $\mathcal{D}$  and for each solution  $D \in \mathcal{D}$  which satisfies:

$$MSB_u(KBRW_K(M_R) \oplus KBRW_K(M_R \oplus D)) = 0^u,$$

where  $u \leq \tau$ .

3. We query the decryption of DCT with  $(C_L \| C_R \oplus D)$  and observe the output to determine whether the forgery attack is successful or not.  
If the decryption output is  $\perp$ , we select a new difference  $D$  from  $\mathcal{D}$  and make the same decryption queries. We repeat the preceding step approximately  $2^{\tau-u}$  times until we obtain a successful forgery that the decryption output does not equal  $\perp$ .

Since the associated data has the same effect as the message, the attacks discussed in the following sections still work when  $A \neq \varepsilon$ .

## 5.2 An Attack to DCT with Fixed Message Length

Assume adversaries query the KBRW function with a message of length  $m = 2^u - 1$ . When  $u = 2$ ,  $M = (M_1, M_2, M_3)$ , consider

$$KBRW_K(M) = K^4 \oplus M_1 K^3 \oplus M_2 K^2 \oplus (M_1 M_2 \oplus M_3) K, \quad (1)$$

let  $M_1$  remain constant ( $D_1 = 0$ ), and only modify  $M_2$  and  $M_3$  by unknowns  $D_2$  and  $D_3$ , respectively, so that  $KBRW_K(M) \oplus KBRW_K(M \oplus D) = D_2 K^2 \oplus D_3 K$  is a linear function of  $K$ , where  $D = (D_1, D_2, D_3)$ , and we can calculate  $D_2$  and  $D_3$  by the technique outlined in Section 3.

When  $u = 3$ , the situation becomes complicated, consider

$$\begin{aligned} KBRW_K(M) = & K^8 \oplus M_1K^7 \oplus M_2K^6 \oplus (M_1M_2 \oplus M_3)K^5 \\ & \oplus (M_4 \oplus 1)K^4 \oplus (M_1M_4 \oplus M_5)K^3 \oplus (M_2M_4 \oplus M_6)K^2 \\ & \oplus (M_1M_2M_4 \oplus M_3M_4 \oplus M_5M_6 \oplus M_7)K. \end{aligned} \quad (2)$$

We choose  $M_4$ ,  $M_6$ , and  $M_7$  as blocks modified by unknowns. Still, some message blocks cannot be chosen arbitrarily: For example, the coefficient of  $K^4$  is  $M_4 \oplus 1$ , and the coefficient of  $K^3$  is  $M_1M_4 \oplus M_5$ ; To obtain the final linear function,  $D_1$  must be chosen as 0.

For  $u \geq 2$ , to linearize  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$ , we divide the message blocks into six categories of disjoint set:  $\mathcal{V}_0^u$ ,  $\mathcal{V}_1^u$ ,  $\mathcal{A}_0^u$ ,  $\mathcal{A}_1^u$ ,  $\mathcal{F}_0^u$  and  $\mathcal{F}_1^u$  for the KBRW function, which specifies how the input messages should be valued.

- $\mathcal{V}_0^u$  and  $\mathcal{V}_1^u$  are sets of blocks that can be chosen arbitrarily and modified by unknowns. If the block  $M_i$  appears as the coefficient of a power term in the form of  $M_i \oplus 1$ , we put it in  $\mathcal{V}_1^u$ , otherwise in  $\mathcal{V}_0^u$ ;
- $\mathcal{A}_0^u$  and  $\mathcal{A}_1^u$  are sets of blocks that can be chosen arbitrarily but not modified by unknowns. The difference between  $\mathcal{A}_0^u$  and  $\mathcal{A}_1^u$  is the same as the above;
- $\mathcal{F}_0^u$  and  $\mathcal{F}_1^u$  are sets of blocks that are fixed as 0 and 1 respectively and not modified by unknowns. The difference between  $\mathcal{F}_0^u$  and  $\mathcal{F}_1^u$  is the same as above.

For example, when  $u = 2$ , consider Equation (1). Let  $\mathcal{F}_0^2 = \mathcal{F}_1^2 = \mathcal{V}_1^2 = \mathcal{A}_1^2 = \emptyset$ ,  $\mathcal{V}_0^2 = \{M_2, M_3\}$  and  $\mathcal{A}_0^2 = \{M_1\}$ , which means that the value of  $M_1$  should remain constant in our forgery attacks ( $D_1 = 0$ ); And we can calculate  $D_2$  and  $D_3$  so that  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$  is a linear function of  $K$ .

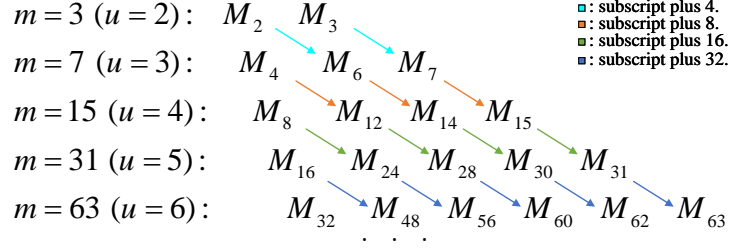
For  $u > 2$ , we intend to obtain these sets recursively. Equation (2) can also be denoted as:

$$KBRW_K(M) = KBRW_K(M_1, M_2, M_3)(K^4 \oplus M_4) \oplus KBRW_K(M_5, M_6, M_7).$$

Similarly to the analysis of  $KBRW_K(M_1, M_2, M_3)$ , in  $KBRW_K(M_5, M_6, M_7)$  we choose  $\mathcal{V}_0^3$  by the case with  $u = 2$ , and they originate from the subscript plus 4 of elements in  $\mathcal{V}_0^2$ , as shown in Figure 2, so that  $\mathcal{V}_0^3 = \{M_{i+2^2} | M_i \in \mathcal{V}_0^2\} = \{M_6, M_7\}$ . Thus, for general  $u > 2$ , we have  $\mathcal{V}_0^u = \{M_{i+2^{u-1}} | M_i \in \mathcal{V}_0^{u-1}\}$ .

Consider the general case when  $u \geq 2$ , we formalize the above deduction as the following Theorem 2.

**Theorem 2.** *For the KBRW function, assume  $m = 2^u - 1$ ,  $u \geq 2$ . Let  $\mathcal{V}_0^2 = \{M_2, M_3\}$ ,  $\mathcal{A}_0^2 = \{M_1\}$ , and initialize the remaining set to  $\emptyset$ . We can obtain the*



**Fig. 2.** Calculating the set  $\mathcal{V}_0^u$  and  $\mathcal{V}_1^u$  recursively, where the arrow indicates that the subscript plus  $t$ .

following recursions:

$$\begin{aligned}
\mathcal{V}_0^u &= \{M_{i+2^{u-1}} | M_i \in \mathcal{V}_0^{u-1}\}, \\
\mathcal{V}_1^u &= \{M_{2^{u-1}}\} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{V}_1^{u-1}\}, \\
\mathcal{A}_0^u &= \mathcal{V}_0^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{A}_0^{u-1}\}, \\
\mathcal{A}_1^u &= \mathcal{V}_1^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{A}_1^{u-1}\}, \\
\mathcal{F}_0^u &= \mathcal{F}_0^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_0^{u-1}\} \cup \mathcal{A}_0^{u-1}, \\
\mathcal{F}_1^u &= \mathcal{F}_1^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_1^{u-1}\} \cup \mathcal{A}_1^{u-1},
\end{aligned}$$

where  $i \in \mathbb{Z}^+$ . Then, after assigning the message blocks according to the recursions above,  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$  is a linear function of  $K$ .

*Proof.* When  $u = 3$ , by analyzing  $KBRW_K(M_1, \dots, M_7)$ , we can obtain the following conclusions:

$$\begin{aligned}
\mathcal{V}_0^3 &= \{M_{i+2^2} | M_i \in \mathcal{V}_0^2\} = \{M_6, M_7\}, \\
\mathcal{V}_1^3 &= \{M_{2^2}\} \cup \{M_{i+2^2} | M_i \in \mathcal{V}_1^2\} = \{M_4\}, \\
\mathcal{A}_0^3 &= \mathcal{V}_0^2 \cup \{M_{i+2^2} | M_i \in \mathcal{A}_0^2\} = \{M_2, M_3, M_5\}, \\
\mathcal{A}_1^3 &= \mathcal{V}_1^2 \cup \{M_{i+2^2} | M_i \in \mathcal{A}_1^2\} = \emptyset, \\
\mathcal{F}_0^3 &= \mathcal{F}_0^2 \cup \{M_{i+2^2} | M_i \in \mathcal{F}_0^2\} \cup \mathcal{A}_0^2 = \{M_1\}, \\
\mathcal{F}_1^3 &= \mathcal{F}_1^2 \cup \{M_{i+2^2} | M_i \in \mathcal{F}_1^2\} \cup \mathcal{A}_1^2 = \emptyset,
\end{aligned}$$

which means when  $M_2, M_3$  and  $M_5$  remain constant ( $D_2 = D_3 = D_5 = 0$ ), and let  $M_1 = 0$ , we can calculate  $D_4, D_6$  and  $D_7$  using the linear modification technique stated in Section 3 such that  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$  is a linear function of  $K$ . Therefore, the conclusion is true.

Suppose that the conclusion is true in the case  $u - 1$ . Next, we consider the KBRW function in case  $u$ :

$$\begin{aligned} KBRW_K(M_1, \dots, M_m) &= KBRW_K(M_1, \dots, M_{2^{u-1}-1})(K^{2^{u-1}} \oplus M_{2^{u-1}}) \\ &\oplus KBRW_K(M_{2^{u-1}+1}, \dots, M_m). \end{aligned} \quad (3)$$

Since the subscript of coefficients in  $KBRW_K(M_{2^{u-1}+1}, \dots, M_m)$  is larger than  $KBRW_K(M_1, \dots, M_{2^{u-1}-1})$  by  $2^{u-1}$ , therefore, as shown in Figure 2, we have  $\mathcal{V}_0^u = \{M_{i+2^{u-1}} | M_i \in \mathcal{V}_0^{u-1}\}$  and  $\{M_{i+2^{u-1}} | M_i \in \mathcal{V}_1^{u-1}\} \subseteq \mathcal{V}_1^u$ .

To linearize the above two functions, we need to assign 0 and 1 to each block in  $\mathcal{F}_0^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_0^{u-1}\}$  and  $\mathcal{F}_1^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_1^{u-1}\}$ , respectively. Thus, we have:

$$\begin{aligned} \mathcal{F}_0^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_0^{u-1}\} &\subseteq \mathcal{F}_0^u, \\ \mathcal{F}_1^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_1^{u-1}\} &\subseteq \mathcal{F}_1^u. \end{aligned}$$

Assume the linearized Equation (3) can be denoted as:

$$\begin{aligned} L_K(M_1, \dots, M_m) &= L_K(M_1, \dots, M_{2^{u-1}-1})(K^{2^{u-1}} \oplus M_{2^{u-1}}) \\ &\oplus L_K(M_{2^{u-1}+1}, \dots, M_m), \end{aligned} \quad (4)$$

the leading term of  $L_K(M_1, \dots, M_{2^{u-1}-1})M_{2^{u-1}}$  and  $L_K(M_{2^{u-1}+1}, \dots, M_m)$  are  $M_{2^{u-1}}K^{2^{u-1}}$  and  $K^{2^{u-1}}$ , respectively. Thus, the coefficient of  $K^{2^{u-1}}$  in Equation (4) is  $M_{2^{u-1}} \oplus 1$ . Therefore, we have  $\mathcal{V}_0^u = \{M_{2^{u-1}}\} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{V}_1^{u-1}\}$ .

We continue linearizing Equation (4). Since  $L_K(M_1, \dots, M_{2^{u-1}-1})$  times  $K^{2^{u-1}}$  raises the degree of each term by  $2^{u-1}$ , so none of the degrees of each term in this function are powers of 2 except the leading term  $K^{2^u}$ . To linearize it, each block in  $\mathcal{V}_0^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{A}_0^{u-1}\}$  and  $\mathcal{V}_1^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{A}_1^{u-1}\}$  should keep unchanged when forgery. Thus, we have:

$$\begin{aligned} \mathcal{A}_0^u &= \mathcal{V}_0^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{A}_0^{u-1}\}, \\ \mathcal{A}_1^u &= \mathcal{V}_1^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{A}_1^{u-1}\}. \end{aligned}$$

Because  $M_{2^{u-1}} \in \mathcal{V}_1^u$ , when linearizing  $L_K(M_1, \dots, M_{2^{u-1}-1})M_{2^{u-1}}$ , we need to assign each block in  $\mathcal{A}_0^{u-1}$  and  $\mathcal{A}_1^{u-1}$  as 0 or 1, respectively. So we have:

$$\begin{aligned} \mathcal{F}_0^u &= \mathcal{F}_0^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_0^{u-1}\} \cup \mathcal{A}_0^{u-1}, \\ \mathcal{F}_1^u &= \mathcal{F}_1^{u-1} \cup \{M_{i+2^{u-1}} | M_i \in \mathcal{F}_1^{u-1}\} \cup \mathcal{A}_1^{u-1}. \end{aligned}$$

Therefore, the conclusion is true in case  $u$ . □

After linearizing the KBRW function using the technique stated in the above Theorem 2, we can create a linear system  $\mathcal{S}$  containing  $u \times 128$  linear equations and  $(u + 1) \times 128$  unknowns to calculate a solution set  $\mathcal{D}$  using the technique





$MSB_{16}(\mathcal{H}_{K_1\|K_2}(A, M_R)) = MSB_{16}(\mathcal{H}_{K_1\|K_2}(A, M_R \oplus D))$  for each  $D \in \mathcal{D}$ . Third, adversaries continue to make  $2^{16}$  decryption queries for the modified ciphertext  $(C_L\|C_R \oplus D)$  by selecting a new difference  $D$  from  $\mathcal{D}$  to match the remaining 16 bits of the stretch. Note that adversaries need to perform about  $q = \mathcal{O}(2^{16})$  queries to obtain a successful forgery. Therefore, our attack can fail the integrity with fewer queries when  $\tau$  is small.

### 5.3 An attack to DCT with Arbitrary Length Message

In Section 5.2, we fix the message length that the adversaries query the KBRW function to  $m = 2^u - 1$ , which is a particular type of attack that does not have strong universality. Users usually select arbitrary-length messages to query the scheme. Therefore, this section considers how to linearize the KBRW function with arbitrary-length messages.

For arbitrary  $m$ , we define six categories of disjoint sets of message blocks as  $V_0^m, V_1^m, A_0^m, A_1^m, F_0^m$  and  $F_1^m$  for the KBRW function, they are similar to the sets defined in Section 5.2, except that the former targets messages of arbitrary length while the latter targets messages of length  $m = 2^u - 1$ . Next, we generalize the above conclusion to Theorem 3.

**Theorem 3.** *For the KBRW function, assuming the message length is  $m$ ,  $t \leq m < 2t$ ,  $t = 2^u$ ,  $u \geq 2$ . Let  $V_0^1 = \{M_1\}$ ,  $V_0^2 = \{M_1, M_2\}$ ,  $V_0^3 = \{M_2, M_3\}$ ,  $A_0^3 = \{M_1\}$  and initialize the remaining set to  $\emptyset$ . We can obtain the following recursions when  $m \geq 4$ :*

$$\begin{aligned} A_0^m &= V_0^{t-1} \bigcup \{M_{i+t} | M_i \in A_0^{m-t}\}, \\ A_1^m &= V_1^{t-1} \bigcup \{M_{i+t} | M_i \in A_1^{m-t}\}, \\ F_0^m &= F_0^{t-1} \bigcup \{M_{i+t} | M_i \in F_0^{m-t}\} \bigcup A_0^{t-1}, \\ F_1^m &= F_1^{t-1} \bigcup \{M_{i+t} | M_i \in F_1^{m-t}\} \bigcup A_1^{t-1}. \end{aligned}$$

And we can obtain the following recursions when  $m \geq 7$ :

$$\begin{aligned} V_0^m &= \begin{cases} \{M_{i+t} | M_i \in V_0^{m-t}\} \bigcup \{M_t\}, & m < \frac{3t}{2} \\ \{M_{i+t} | M_i \in V_0^{m-t}\}, & \text{otherwise} \end{cases} \\ V_1^m &= \begin{cases} \{M_{i+t} | M_i \in V_1^{m-t}\}, & m < \frac{3t}{2} \\ \{M_{i+t} | M_i \in V_1^{m-t}\} \bigcup \{M_t\}, & \text{otherwise,} \end{cases} \end{aligned}$$

where  $i \in \mathbb{Z}^+$ . Then, after assigning the message blocks according to the above recursions,  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$  is a linear function of  $K$ .

*Proof.* Assuming the linearized KBRW function can be denoted as:

$$\begin{aligned} L_K(M_1, \dots, M_m) &= L_K(M_1, \dots, M_{t-1})(K^t \oplus M_t) \\ &\quad \oplus L_K(M_{t+1}, \dots, M_m), \quad t \leq m < 2t. \end{aligned} \tag{7}$$

However,  $K^{2t}$  is the leading term of  $KBRW_K(M_1, \dots, M_{2^u})$ ,  $t = 2^u, u \geq 2$ . Therefore, whether the coefficient of  $K^t$  is  $M_t \oplus 1$  or  $M_t$  in Equation (7) depends on whether the leading term of  $L_K(M_{t+1}, \dots, M_m)$  is  $K^t$  or not, and it's true only when  $m - t \geq t/2$ . Then we have  $\{M_t\} \subseteq F_1^m$ , otherwise  $\{M_t\} \subseteq F_0^m$ . Thus, when  $m \geq 7$ , we have:

$$V_0^m = \begin{cases} \{M_{i+t}|M_i \in V_0^{m-t}\} \cup \{M_t\}, & m < \frac{3t}{2} \\ \{M_{i+t}|M_i \in V_0^{m-t}\}, & otherwise \end{cases}$$

$$V_1^m = \begin{cases} \{M_{i+t}|M_i \in V_1^{m-t}\}, & m < \frac{3t}{2} \\ \{M_{i+t}|M_i \in V_1^{m-t}\} \cup \{M_t\}, & otherwise. \end{cases}$$

The rest of the proof is the same as Theorem 2 and will not repeat here.  $\square$

As we can see, Theorem 2 is a particular case of Theorem 3. Next, we analyze our attack's complexity by finding the smallest  $m$  satisfies  $|V_0^m| + |V_1^m| = u$  for a fixed  $u, u \geq 2$ .

**Theorem 4.** *For the KBRW function,  $m = 2^u - 2$  is the smallest message length that satisfies  $|V_0^m| + |V_1^m| = u, u \geq 2$ . Then, after assigning the message blocks according to the above recursions,  $KBRW_K(M) \oplus KBRW_K(M \oplus D)$  is a linear function of  $K$ .*

*Proof.*

Define  $V^m = V_0^m \cup V_1^m$ , we have  $|V^2| = |V^3| = 2$ . Therefore,  $m = 2^2 - 2 = 2$  is the minimal message length in case  $u = 2$ .

According to Theorem 3, we have  $|V^m| = |V^{m-t}| + 1$ . To obtain one more block that can be chosen arbitrarily and modified by unknowns, set  $m - t = 2$ , solving for the smallest  $m$  and  $t$  as 6 and 4, respectively. Therefore,  $m = 2^3 - 2 = 6$  is the minimal message length in case  $u = 3$ .

Assume  $m = 2^u - 2$  is the minimal message length in case  $u$ . To obtain one more block that we want, let  $m' - t' = m$  and solve this equation:

$$m' - t' = 2^{u+1} - 2 - 2^u = 2^u - 2 = m,$$

the smallest  $m'$  and  $t'$  are  $2^{u+1} - 2$  and  $2^u$ , respectively. Therefore,  $m' = 2^{u+1} - 2$  is the optimal solution in case  $u + 1$ .  $\square$

For a particular  $m = 2^u - 2, u \geq 2$ , define new sets  $A^m = A_0^m \cup A_1^m$ , and  $F^m = F_0^m \cup F_1^m$ , and we can calculate the size of  $V^m, A^m$  and  $F^m$  by the following expression:

$$V(m) = u, A(m) = \sum_{i=2}^u i, F(m) = m - u - \sum_{i=2}^u i.$$

Similarly, we can convert this improved attack on the KBRW function into an attack on  $\mathcal{H}$ . Adversaries can select a message of  $m = 2^{u+2} - 3$  blocks and successfully execute our attack with  $q = 2^v$  decryption queries ( $u + v = \tau$ ). Like the attack in Section 3, this technique can efficiently balance the space and time complexity required by the attack by flexibly choosing the values of  $m$  and  $q$ . Moreover, it is possible to recover the authentication key using the technique described in Section 5 of [9], which would compromise DCT's confidentiality.

#### 5.4 Analysis of the Security Bounds with DCT Instantiation Scheme

In this section, we analyze the security bounds of DCT. Let  $\tilde{\Pi}$  denote the DCT scheme and let  $\tilde{E}$  denote the tweakable blockcipher used by the encryption scheme  $\Pi_1$ ,  $K \xleftarrow{\$} \mathcal{K}$ . Theorem 3 in [10] describes the DAE security of  $\tilde{\Pi}$ . Let  $\mathbf{A}$  be an adversary on  $\tilde{\Pi}$  that asks at most  $q$  queries of at most  $m$  blocks in total and runs in time at most  $t$ . Then, the  $\mathbf{Adv}_{\tilde{\Pi}}^{\text{DAE}}(\mathbf{A})$  is upper bounded by

$$\frac{3q^2\epsilon}{2} + \frac{2q^2}{2^{2n}} + \frac{3q\epsilon \cdot 2^{2n}}{2^\tau} + 3 \cdot \mathbf{Adv}_{\tilde{E}, \tilde{E}^{-1}}^{\text{SPRP}}(q, \mathcal{O}(t)) + 2 \cdot \mathbf{Adv}_{\Pi_1}^{\text{ivE}}(q, m, \mathcal{O}(t)).$$

For small  $\tau$ , the security of DCT depends on the leading term  $\frac{q\epsilon \cdot 2^{2n}}{2^\tau}$  mostly, which is related to not only  $q$  but  $\epsilon$ . When DCT are implemented with BRW Hashing, i.e.,  $\epsilon = \mathcal{O}(\frac{m^2}{2^{2n}})$  [10], the provable bounds of DCT are  $\mathcal{O}(\frac{q^2 m^2}{2^{2n}} + \frac{qm^2}{2^\tau})$ , and we only focus on the term  $\frac{qm^2}{2^\tau}$ . Let  $u + v = \tau$ , when adversaries make  $q = \mathcal{O}(2^v)$  decryption queries of  $m = \mathcal{O}(2^{u+2})$  blocks,  $\frac{qm^2}{2^\tau} > 1$ . We remark that the security bounds between the above attack and the proof are not contradictory.

As a result, the above analysis shows that DCT has a similar problem as GCM. The attack succeeds because the stretch part no longer holds the  $\epsilon$ -AXU property well when  $\tau$  is small. Furthermore, it can attack DCT's confidentiality by recovering the authentication key using the technique described in Section 5 of [9]. In the following section, we introduce Robust DCT (RDCT), a variant of the DCT scheme in which the security bound is better than DCT, and the above attack is invalid for RDCT.

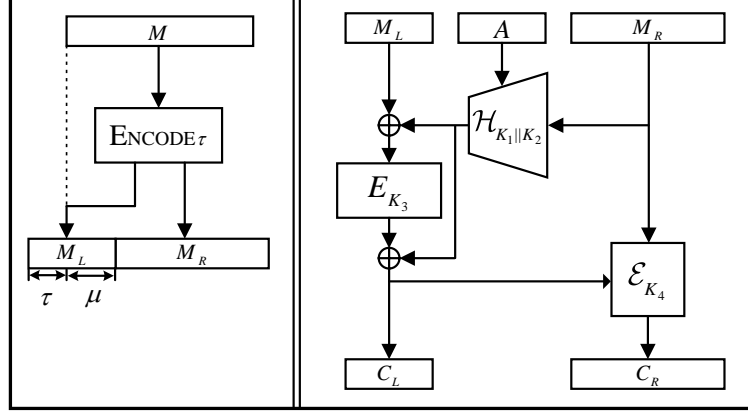
## 6 How to Fix It: Robust DCT

Our attack works due to the way DCT deals with the stretch. DCT encrypts  $M_L$  by XORing it with the result of  $\mathcal{H}$ , which does not prevent manipulation of the stretch. To make  $M_L$  unpredictable, in this section, we slightly modify DCT to avoid the problem in Section 5 by simply XORing the output of  $\mathcal{H}$  to the output of keyed permutation  $E$ . Therefore, encryption (resp. decryption) queries to it will lead to a random left output. We call the new scheme Robust DCT (RDCT). We show the encryption of RDCT scheme  $\text{RDCT}_{\mathcal{H}, E, \Pi_1, \Pi_2} = (\tilde{\mathcal{E}}, \tilde{\mathcal{D}})$  in Figure 3 and the RDCT scheme in Algorithm 2.

In fact, the modification forms a tweakable blockcipher  $\tilde{E}$  based on  $\mathcal{H}_{K_1 \| K_2}$  and  $E_{K_3}$ :

$$\tilde{E}_{K_1, K_2, K_3}((A, M_R), M_L) := E_{K_3}(M_L \oplus \mathcal{H}_{K_1 \| K_2}(A, M_R)) \oplus \mathcal{H}_{K_1 \| K_2}(A, M_R). \quad (8)$$

The idea is similar to the paper by Ashur et al. [1], which introduces minor tweaks, such as an additional XOR, to get a tweakable blockcipher. Moreover, as a result, we get RDCT, which is an instantiation of UIV construction [6].



**Fig. 3.** The  $\text{ENCODE}_\tau$  process (left). The encryption process of RDCT (right).

Assuming all queries are different and adversaries do not make the output of encryption (resp. decryption) queries to decryption (resp. encryption) queries.  $\tilde{E}$  is an STPRP, then encryption (resp. decryption) queries to RDCT will lead to a random left output  $C_L$  (resp.  $M_L$ ). This modification will enhance the confidentiality of the scheme by changing the integrity promised from the  $\epsilon$ -AXU property of  $\mathcal{H}$  to the randomness of  $M_L$ , which will be recovered from decryption queries. This modification also makes the stretch length in integrity independent of the maximal number of message blocks.

---

**Algorithm 2** Encryption and decryption of the RDCT construction

---

```

1: function  $\tilde{\mathcal{E}}_{K_1, K_2, K_3, K_4}(A, M)$ 
2:    $(M_L, M_R) \leftarrow \text{ENCODE}_\tau(M)$ 
3:    $C_L \leftarrow E_{K_3}(M_L \oplus \mathcal{H}_{K_1\|K_2}(A, M_R)) \oplus \mathcal{H}_{K_1\|K_2}(A, M_R)$ 
4:    $C_R \leftarrow \mathcal{E}_{K_4}(C_L, M_R)$ 
5:   return  $(C_L\|C_R)$ 
6: end function
7:
8: function  $\tilde{\mathcal{D}}_{K_1, K_2, K_3, K_4}(A, C)$ 
9:    $(C_L, C_R) \leftarrow C$ 
10:   $M_R \leftarrow \mathcal{D}_{K_4}(C_L, C_R)$ 
11:   $M_L \leftarrow E_{K_3}^{-1}(C_L \oplus \mathcal{H}_{K_1\|K_2}(A, M_R)) \oplus \mathcal{H}_{K_1\|K_2}(A, M_R)$ 
12:  return  $\text{DECODE}_\tau(M_L\|M_R)$ 
13: end function

```

---

In the following, we show the security bounds of the RDCT scheme.

**Lemma 1 (Confidentiality Advantage of RDCT).** *Let  $\tilde{\Pi} = \text{RDCT}_{\mathcal{H},E,\Pi_1,\Pi_2}$  be as defined in Algorithm 2. Let  $\mathbf{A}$  be a  $\text{DETPRIV}$  adversary on  $\tilde{\Pi}$  that submits at most  $q_e$  encryption queries of at most  $m$  blocks in total and runs in time at most  $t$ . Then*

$$\text{Adv}_{\tilde{\Pi}}^{\text{DETPRIV}}(\mathbf{A}) \leq 3q_e^2\epsilon + \frac{q_e(q_e - 1)}{2^{2n}} + \text{Adv}_E^{\text{PRP}}(q_e, \mathcal{O}(t+q_e)) + \text{Adv}_{\Pi_1}^{\text{IVe}}(q_e, m, \mathcal{O}(t)).$$

**Lemma 2 (Integrity Advantage of RDCT).** *Let  $\tilde{\Pi} = \text{RDCT}_{\mathcal{H},E,\Pi_1,\Pi_2}$  be as defined in Algorithm 2. Let  $\mathbf{A}$  be a  $\text{DETAUTH}$  adversary on  $\tilde{\Pi}$  that submits at most  $q_e$  encryption queries and  $q_d$  decryption queries of at most  $m$  blocks in total, and runs in time at most  $t$ . Then*

$$\text{Adv}_{\tilde{\Pi}}^{\text{DETAUTH}}(\mathbf{A}) \leq 3q^2\epsilon + \frac{q}{2^\tau - q} + \text{Adv}_{E,E^{-1}}^{\text{SPRP}}(q, \mathcal{O}(t+q))$$

where  $q = q_e + q_d$ .

Lemma 1 and 2 are proved in Appendices A and B, respectively.

**Theorem 5 (DAE Advantage of RDCT).** *Let  $\tilde{\Pi} = \text{RDCT}_{\mathcal{H},E,\Pi_1,\Pi_2}$  be as defined in Algorithm 2. Let  $\mathbf{A}$  be a DAE adversary on  $\tilde{\Pi}$  that asks at most  $q_e$  encryption queries and  $q_d$  decryption queries of at most  $m$  blocks in total and runs in time at most  $t$ . Then,  $\text{Adv}_{\tilde{\Pi}}^{\text{DAE}}(\mathbf{A})$  is upper bounded by*

$$\text{Adv}_{\tilde{\Pi}}^{\text{DAE}}(\mathbf{A}) \leq 6q^2\epsilon + \frac{q^2}{2^{2n}} + \frac{q}{2^\tau - q} + 2\text{Adv}_{E,E^{-1}}^{\text{SPRP}}(q, \mathcal{O}(t+q)) + \text{Adv}_{\Pi}^{\text{IVe}}(q, m, \mathcal{O}(t))$$

where  $q = q_e + q_d$ .

The proof of Theorem 5 follows from Theorem 1 and the individual bounds for the  $\text{DETPRIV}$  and  $\text{DETAUTH}$  security in Lemma 1 and 2. For small  $\tau$ , the security of RDCT depends on the leading term  $\frac{q}{2^\tau - q}$  mostly. At this point, it is independent of  $\epsilon$ . However, the provable security of DCT is  $\mathcal{O}(\frac{q^2\epsilon}{2} + \frac{q^2}{2^{2n}} + \frac{q\epsilon \cdot 2^{2n}}{2^\tau})$  [10]. For small  $\tau$ , it depends on the leading term  $\frac{q\epsilon \cdot 2^{2n}}{2^\tau}$  mostly. So it is related to not only  $q$  but  $\epsilon$ . We will show how it affects when DCT and RDCT are implemented with BRW hashing, that is,  $\epsilon = \mathcal{O}(\frac{m^2}{2^{2n}})$  [10]. Now the provable securities of DCT and RDCT are  $\mathcal{O}(\frac{q^2 m^2}{2^{2n}} + \frac{qm^2}{2^\tau})$  and  $\mathcal{O}(\frac{q^2 m^2}{2^{2n}} + \frac{q}{2^\tau - q})$  respectively. For small  $\tau$ , we focus only on  $\frac{qm^2}{2^\tau}$  and  $\frac{q}{2^\tau - q}$  respectively. Note that the security of DCT depends on the length of the query. However, the security of RDCT is almost unaffected by how long adversaries make the query.

The attack to DCT only requires  $\mathcal{O}(2^v)$  queries with  $\mathcal{O}(2^u)$  blocks, where  $v \leq \tau$  and  $u = \tau - v$ . Compared with DCT, to break the integrity of RDCT, adversaries have to make  $\mathcal{O}(2^\tau)$  queries. Therefore, the minor change made by RDCT improves the security of DCT without sacrificing efficiency.

## 7 Conclusions

In this paper, we find that DCT suffers from a small stretch problem similar to that of GCM. Although the BRW polynomial function is more complicated than GHASH, Ferguson’s linear modification technique still works. To obtain a successful forgery of the DCT scheme, our attack must choose the length of the message  $m$  and the number of queries  $q$  flexibly with trade-off  $mq = \mathcal{O}(2^\tau)$  for small  $\tau$ . Both GCM and DCT use the Wegman-Carter [31, 32] framework to perform authentication. If we replace the GHASH function in GCM with KBRW, or use KBRW in Wegman-Carter MAC, our method still works.

To solve the above small stretch problem, we propose an improved scheme named Robust DCT (RDCT) without sacrificing efficiency, improving the security when  $\tau$  is small. The security bound of RDCT is  $q = \mathcal{O}(2^\tau)$ , which demonstrates that our enhancements are successful. Our fixing method is similar to that of [1] to boost the security of GCM: XORing the result of the AXU function both before and after the blockcipher. The core of resulting RDCT actually uses the UIV enciphering algorithm, which is one of the rugged pseudorandom permutations suggested by Degabriele et al. [6].

## References

1. Ashur, T., Dunkelman, O., Luykx, A.: Boosting authenticated encryption robustness with minimal modifications. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 3–33. Springer (2017). [https://doi.org/10.1007/978-3-319-63697-9\\_1](https://doi.org/10.1007/978-3-319-63697-9_1), [https://doi.org/10.1007/978-3-319-63697-9\\_1](https://doi.org/10.1007/978-3-319-63697-9_1) 19, 22
2. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. In: FOCS ’97. pp. 394–403. IEEE Computer Society (1997). <https://doi.org/10.1109/SFCS.1997.646128>, <https://doi.org/10.1109/SFCS.1997.646128> 5
3. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer (2006). [https://doi.org/10.1007/11761679\\_25](https://doi.org/10.1007/11761679_25), [https://doi.org/10.1007/11761679\\_25](https://doi.org/10.1007/11761679_25) 26
4. Bernstein, D.J.: Polynomial evaluation and message authentication. URL: <https://cr.ypt.to/antiforgery/pema-20071022.pdf>. Citations in this document 2 (2007) 2, 3
5. Chakraborty, D., Ghosh, S., Sarkar, P.: A Fast Single-Key Two-Level Universal Hash Function. ToSC 2017(1), 106–128 (2017). <https://doi.org/10.13154/tosc.v2017.i1.106-128>, <https://doi.org/10.13154/tosc.v2017.i1.106-128> 3
6. Degabriele, J.P., Karadzic, V.: Overloading the nonce: Rugged prps, nonce-set aead, and order-resilient channels. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13510, pp. 264–295. Springer (2022). [https://doi.org/10.1007/978-3-031-15985-5\\_10](https://doi.org/10.1007/978-3-031-15985-5_10), [https://doi.org/10.1007/978-3-031-15985-5\\_10](https://doi.org/10.1007/978-3-031-15985-5_10) 2, 19, 22
7. Dworkin, M.J.: SP 800-38D. recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC. NIST (2007) 2

8. Ferguson, N.: Collision attacks on OCB. Comments submitted to NIST Modes of Operation Process pp. 1–13 (2002) [2](#)
9. Ferguson, N.: Authentication weaknesses in GCM. Comments submitted to NIST Modes of Operation Process pp. 1–19 (2005) [2](#), [3](#), [7](#), [18](#), [19](#)
10. Forler, C., List, E., Lucks, S., Wenzel, J.: Efficient beyond-birthday-bound-secure deterministic authenticated encryption with minimal stretch. In: ACISP 2016. LNCS, vol. 9723, pp. 317–332. Springer (2016). [https://doi.org/10.1007/978-3-319-40367-0\\_20](https://doi.org/10.1007/978-3-319-40367-0_20), [https://doi.org/10.1007/978-3-319-40367-0\\_20](https://doi.org/10.1007/978-3-319-40367-0_20) [2](#), [3](#), [8](#), [9](#), [11](#), [19](#), [21](#)
11. Ghosh, S., Sarkar, P.: Evaluating Bernstein-Rabin-Winograd Polynomials. DCC **87**(2-3), 527–546 (2019). <https://doi.org/10.1007/s10623-018-0561-7>, <https://doi.org/10.1007/s10623-018-0561-7> [3](#)
12. Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: Specification and Analysis. Cryptology ePrint Archive (2017) [2](#)
13. Gueron, S., Lindell, Y.: GCM-SIV: Full Nonce Misuse-Resistant Authenticated Encryption at Under One Cycle per Byte. In: CCS. pp. 109–119 (2015) [2](#)
14. Halevi, S., Rogaway, P.: A parallelizable enciphering mode. In: CT-RSA 2004. LNCS, vol. 2964, pp. 292–304. Springer (2004). [https://doi.org/10.1007/978-3-540-24660-2\\_23](https://doi.org/10.1007/978-3-540-24660-2_23), [https://doi.org/10.1007/978-3-540-24660-2\\_23](https://doi.org/10.1007/978-3-540-24660-2_23) [26](#)
15. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: CRYPTO '88. LNCS, vol. 403, pp. 8–26. Springer (1988). [https://doi.org/10.1007/0-387-34799-2\\_2](https://doi.org/10.1007/0-387-34799-2_2), [https://doi.org/10.1007/0-387-34799-2\\_2](https://doi.org/10.1007/0-387-34799-2_2) [26](#)
16. ISO, IEC: ISO/IEC 19772:2020 Information security — Authenticated encryption. ISO (2020) [2](#)
17. Iwata, T., Yasuda, K.: BTM: A Single-Key, Inverse-Cipher-Free Mode for Deterministic Authenticated Encryption. In: SAC 2009. LNCS, vol. 5867, pp. 313–330. Springer (2009). [https://doi.org/10.1007/978-3-642-05445-7\\_20](https://doi.org/10.1007/978-3-642-05445-7_20), [https://doi.org/10.1007/978-3-642-05445-7\\_20](https://doi.org/10.1007/978-3-642-05445-7_20) [2](#)
18. Iwata, T., Yasuda, K.: HBS: A Single-Key Mode of Operation for Deterministic Authenticated Encryption. In: FSE 2009. LNCS, vol. 5665, pp. 394–415. Springer (2009). [https://doi.org/10.1007/978-3-642-03317-9\\_24](https://doi.org/10.1007/978-3-642-03317-9_24), [https://doi.org/10.1007/978-3-642-03317-9\\_24](https://doi.org/10.1007/978-3-642-03317-9_24) [2](#)
19. Jean, J., Nikolic, I., Peyrin, T., Seurin, Y.: Deoxys v1. 41. Submitted to CAESAR **124** (2016) [2](#)
20. Liskov, M.D., Rivest, R.L., Wagner, D.A.: Tweakable Block Ciphers. In: CRYPTO 2002. LNCS, vol. 2442, pp. 31–46. Springer (2002). [https://doi.org/10.1007/3-540-45708-9\\_3](https://doi.org/10.1007/3-540-45708-9_3), [https://doi.org/10.1007/3-540-45708-9\\_3](https://doi.org/10.1007/3-540-45708-9_3) [24](#), [27](#)
21. McGrew, D.A., Viega, J.: The security and performance of the Galois/Counter Mode (GCM) of operation. In: Indocrypt 2004. pp. 343–355. Springer (2004) [2](#)
22. Minematsu, K.: Authenticated encryption with small stretch (or, how to accelerate AERO). In: Liu, J.K., Steinfeld, R. (eds.) ACISP 2016. LNCS, vol. 9723, pp. 347–362. Springer (2016). [https://doi.org/10.1007/978-3-319-40367-0\\_22](https://doi.org/10.1007/978-3-319-40367-0_22), [https://doi.org/10.1007/978-3-319-40367-0\\_22](https://doi.org/10.1007/978-3-319-40367-0_22) [2](#)
23. Peyrin, T., Seurin, Y.: Counter-in-Tweak: Authenticated encryption modes for tweakable block ciphers. In: CRYPTO 2016. LNCS, vol. 9814, pp. 33–63. Springer (2016). [https://doi.org/10.1007/978-3-662-53018-4\\_2](https://doi.org/10.1007/978-3-662-53018-4_2), [https://doi.org/10.1007/978-3-662-53018-4\\_2](https://doi.org/10.1007/978-3-662-53018-4_2) [2](#)
24. Rabin, M.O., Winograd, S.: Fast evaluation of polynomials by rational preparation. CPAM **25**(4), 433–458 (1972) [3](#)

25. Rogaway, P.: Nonce-based symmetric encryption. In: FSE 2004. LNCS, vol. 3017, pp. 348–359. Springer (2004). [https://doi.org/10.1007/978-3-540-25937-4\\_22](https://doi.org/10.1007/978-3-540-25937-4_22), [https://doi.org/10.1007/978-3-540-25937-4\\_22](https://doi.org/10.1007/978-3-540-25937-4_22) 1
26. Rogaway, P., Bellare, M., Black, J.: OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. ACM TISS **6**(3), 365–403 (2003). <https://doi.org/10.1145/937527.937529>, <https://doi.org/10.1145/937527.937529> 2
27. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: EUROCRYPT 2006. LNCS, vol. 4004, pp. 373–390. Springer (2006). [https://doi.org/10.1007/11761679\\_23](https://doi.org/10.1007/11761679_23), [https://doi.org/10.1007/11761679\\_23](https://doi.org/10.1007/11761679_23) 2, 8, 9
28. Rogaway, P., Shrimpton, T.: Deterministic Authenticated-Encryption: A Provable-Security Treatment of the Key-Wrap Problem. Cryptology ePrint Archive, Paper 2006/221 (2006), <https://eprint.iacr.org/2006/221>, <https://eprint.iacr.org/2006/221> 9
29. Sarkar, P.: Efficient tweakable enciphering schemes from (block-wise) universal hash functions. IEEE TIT **55**(10), 4749–4760 (2009). <https://doi.org/10.1109/TIT.2009.2027487>, <https://doi.org/10.1109/TIT.2009.2027487> 3
30. Sarkar, P.: Tweakable enciphering schemes using only the encryption function of a block cipher. Information Processing Letters **111**(19), 945–955 (2011) 3
31. Shoup, V.: On fast and provably secure message authentication based on universal hashing. In: Koblitz, N. (ed.) CRYPTO ’96. Lecture Notes in Computer Science, vol. 1109, pp. 313–328. Springer (1996). [https://doi.org/10.1007/3-540-68697-5\\_24](https://doi.org/10.1007/3-540-68697-5_24), [https://doi.org/10.1007/3-540-68697-5\\_24](https://doi.org/10.1007/3-540-68697-5_24) 22
32. Wegman, M.N., Carter, L.: New hash functions and their use in authentication and set equality. J. Comput. Syst. Sci. **22**(3), 265–279 (1981). [https://doi.org/10.1016/0022-0000\(81\)90033-7](https://doi.org/10.1016/0022-0000(81)90033-7), [https://doi.org/10.1016/0022-0000\(81\)90033-7](https://doi.org/10.1016/0022-0000(81)90033-7) 22

## A Proof of Lemma 1

*Proof.* Let  $\widetilde{\text{Perm}}(\mathcal{X}, \mathcal{Y})$  be a set of all tweakable random permutations on domain/range space  $\mathcal{Y}$  with tweak space  $\mathcal{X}$ , and let  $\widetilde{\text{Func}}(\mathcal{X}, \mathcal{Y})$  be a set of all tweakable random functions on domain/range space  $\mathcal{Y}$  with tweak space  $\mathcal{X}$ . We prove it by the game-playing technique. The games are in Table 2.

$\mathbf{G}_0$ : This is the DETPRIV encryption oracle for the RDCT construction.

$\mathbf{G}_1$ : In the game  $\mathbf{G}_1$ , we replace the tweakable blockcipher  $\widetilde{E}$  in Equation (8) with a tweakable random permutation  $\widetilde{\pi} \xleftarrow{\$} \widetilde{\text{Perm}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}})$ . We bound  $\mathbf{A}$ ’s distinguishing advantage between games  $\mathbf{G}_0$  and  $\mathbf{G}_1$  by constructing an TPRP adversary  $\mathbf{A}_1$  of  $\widetilde{E}$ , where  $\mathbf{A}_1$  submits at most  $q_e$  queries to its oracles and runs in time  $\mathcal{O}(t)$ :

$$|\Pr[\mathbf{A}^{\mathbf{G}_0} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathbf{G}_1} \Rightarrow 1]| \leq \text{Adv}_{\widetilde{E}}^{\text{TPRP}}(q_e, \mathcal{O}(t)). \quad (9)$$

Liskov et al. [20] have proved the STPRP adversary of  $\widetilde{E}$  is bounded by an SPRP adversary  $\mathbf{A}_2$  of  $E$  that submits at most  $q$  queries to its oracles and runs in time



Table 2. Games  $G_0$ - $G_5$ .

<p><b>initialize</b> <span style="float: right;"><math>G_0</math></span></p> <p> <math>K_1 \xleftarrow{\\$} \mathcal{K}_1; K_2 \xleftarrow{\\$} \mathcal{K}_2; K_3 \xleftarrow{\\$} \mathcal{K}_3; K_4 \xleftarrow{\\$} \mathcal{K}_4</math>  <math>\mathcal{X} \leftarrow \emptyset</math> </p> <p> <b>oracle</b> <math>\text{Enc}(A, M)</math>  <b>if</b> <math>(A, M) \in \mathcal{X}</math> <b>then</b>              <b>return</b> <math>\perp</math>  <math>(M_L, M_R) \leftarrow \text{ENCODE}_\tau(M)</math>  <math>C_L \leftarrow E_{K_3}(M_L \oplus \mathcal{H}_{K_1 \parallel K_2}(A, M_R))</math>              <math>\oplus \mathcal{H}_{K_1 \parallel K_2}(A, M_R)</math>  <math>C_R \leftarrow \mathcal{E}_{K_4}(C_L, M_R)</math>  <math>\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}</math>  <b>return</b> <math>(C_L \parallel C_R)</math> </p>	<p><b>initialize</b> <span style="float: right;"><math>G_1</math></span></p> <p> <math>\tilde{\pi} \xleftarrow{\\$} \widetilde{\text{Perm}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}}); K_4 \xleftarrow{\\$} \mathcal{K}_4</math>  <math>\mathcal{X} \leftarrow \emptyset</math> </p> <p> <b>oracle</b> <math>\text{Enc}(A, M)</math>  <b>if</b> <math>(A, M) \in \mathcal{X}</math> <b>then</b>              <b>return</b> <math>\perp</math>  <math>(M_L, M_R) \leftarrow \text{ENCODE}_\tau(M)</math>  <math>C_L \leftarrow \tilde{\pi}((A, M_R), M_L)</math>  <math>C_R \leftarrow \mathcal{E}_{K_4}(C_L, M_R)</math>  <math>\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}</math>  <b>return</b> <math>(C_L \parallel C_R)</math> </p>
<p><b>initialize</b> <span style="float: right;"><math>G_2</math></span></p> <p> <math>\rho \xleftarrow{\\$} \widetilde{\text{Func}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}})</math>  <math>K_4 \xleftarrow{\\$} \mathcal{K}_4</math>  <math>\mathcal{X} \leftarrow \emptyset</math> </p> <p> <b>oracle</b> <math>\text{Enc}(A, M)</math>  <b>if</b> <math>(A, M) \in \mathcal{X}</math> <b>then</b>              <b>return</b> <math>\perp</math>  <math>(M_L, M_R) \leftarrow \text{ENCODE}_\tau(M)</math>  <math>C_L \leftarrow \rho((A, M_R), M_L)</math>  <math>C_R \leftarrow \mathcal{E}_{K_4}(C_L, M_R)</math>  <math>\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}</math>  <b>return</b> <math>(C_L \parallel C_R)</math> </p>	<p><b>initialize</b> <span style="float: right;"><math>G_3</math></span></p> <p> <math>\rho \xleftarrow{\\$} \widetilde{\text{Func}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}})</math>  <math>\rho' \xleftarrow{\\$} \widetilde{\text{Func}}(\{0, 1\}^{2n}, \mathcal{M}_{\mathcal{R}})</math>  <math>\mathcal{X} \leftarrow \emptyset</math> </p> <p> <b>oracle</b> <math>\text{Enc}(A, M)</math>  <b>if</b> <math>(A, M) \in \mathcal{X}</math> <b>then</b>              <b>return</b> <math>\perp</math>  <math>(M_L, M_R) \leftarrow \text{ENCODE}_\tau(M)</math>  <math>C_L \leftarrow \rho((A, M_R), M_L)</math>  <math>C_R \leftarrow \rho'(C_L, M_R)</math>  <math>\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}</math>  <b>return</b> <math>(C_L \parallel C_R)</math> </p>
<p><b>initialize</b> <span style="float: right;"><math>G_4</math></span></p> <p> <math>\rho \xleftarrow{\\$} \widetilde{\text{Func}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}})</math>  <math>\rho' \xleftarrow{\\$} \widetilde{\text{Func}}(\{0, 1\}^{2n}, \mathcal{M}_{\mathcal{R}})</math>  <math>\mathcal{X} \leftarrow \emptyset, \mathcal{Y} \leftarrow \emptyset</math> </p> <p> <b>oracle</b> <math>\text{Enc}(A, M)</math>  <b>if</b> <math>(A, M) \in \mathcal{X}</math> <b>then</b>              <b>return</b> <math>\perp</math>  <math>(M_L, M_R) \leftarrow \text{ENCODE}_\tau(M)</math>  <math>C_L \leftarrow \rho((A, M_R), M_L)</math>  <math>C_R \leftarrow \rho'(C_L, M_R)</math>  <b>if</b> <math>(C_L, M_R) \in \mathcal{Y}</math> <b>then</b>              <math>C_R \xleftarrow{\\$} \{0, 1\}^{ M_R }</math>  <math>\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}</math>  <math>\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(C_L, M_R)\}</math>  <b>return</b> <math>(C_L \parallel C_R)</math> </p>	<p><b>initialize</b> <span style="float: right;"><math>G_5</math></span></p> <p> <math>\mathcal{X} \leftarrow \emptyset</math> </p> <p> <b>oracle</b> <math>\text{Enc}(A, M)</math>  <b>if</b> <math>(A, M) \in \mathcal{X}</math> <b>then</b>              <b>return</b> <math>\perp</math>  <math>(C_L \parallel C_R) \xleftarrow{\\$} \{0, 1\}^{\tau +  M }</math>  <math>\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}</math>  <b>return</b> <math>(C_L \parallel C_R)</math> </p>

$\mathcal{O}(t+q)$ , plus a term  $3q^2\epsilon$  from the  $\epsilon$ -AXU hash function:

$$\mathbf{Adv}_{\widetilde{E}, \widetilde{E}^{-1}}^{\text{STPRP}}(q, \mathcal{O}(t)) \leq \mathbf{Adv}_{E, E^{-1}}^{\text{SPRP}}(q, \mathcal{O}(t+q)) + 3q^2\epsilon,$$

where  $q$  is the whole number of queries to  $\widetilde{E}$  and  $\widetilde{E}^{-1}$ . Thus, it is easy to obtain

$$\mathbf{Adv}_{\widetilde{E}}^{\text{TPRP}}(q_e, \mathcal{O}(t)) \leq \mathbf{Adv}_{E, E^{-1}}^{\text{PRP}}(q_e, \mathcal{O}(t+q_e)) + 3q_e^2\epsilon \quad (10)$$

holds as well.

**G<sub>2</sub>**: In the game **G<sub>2</sub>**, we replace the tweakable random permutation  $\widetilde{\pi}$  with a tweakable random function  $\rho \xleftarrow{\$} \widetilde{\text{Func}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}})$ . We bound **A**'s distinguishing advantage between games **G<sub>1</sub>** and **G<sub>2</sub>** by TPRP-TPRF switching lemma [14]:

$$|\Pr[\mathbf{A}^{\mathbf{G}_1} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathbf{G}_2} \Rightarrow 1]| \leq \frac{q_e(q_e - 1)}{2^{2n+1}}. \quad (11)$$

**G<sub>3</sub>**: In the game **G<sub>3</sub>**, we replace the IV-based encryption algorithm  $\mathcal{E}$  with a tweakable random function  $\rho' \xleftarrow{\$} \widetilde{\text{Func}}(\{0, 1\}^{2n}, \mathcal{M}_{\mathcal{R}})$ . We bound **A**'s distinguishing advantage between games **G<sub>2</sub>** and **G<sub>3</sub>** by constructing an IV-based encryption scheme adversary **A<sub>3</sub>**, where **A<sub>3</sub>** submits at most  $q_e$  queries of at most  $m$  blocks in total to its oracles and runs in time  $\mathcal{O}(t)$ :

$$|\Pr[\mathbf{A}^{\mathbf{G}_2} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathbf{G}_3} \Rightarrow 1]| \leq \mathbf{Adv}_{H_1}^{\text{IVe}}(q_e, m, \mathcal{O}(t)). \quad (12)$$

**G<sub>4</sub>**: In the game **G<sub>4</sub>**, we sample  $C_R$  for repeated tuple  $(C_L, M_R)$ . There are two possibilities in the game **G<sub>3</sub>**. In the first case,  $M_R$  has not appeared before. Then  $(C_L, M_R)$  will not repeat. In the other case, the probability of the repetition of  $C_L$  bounded by PRP-PRF switching lemma [3, 15]. So

$$|\Pr[\mathbf{A}^{\mathbf{G}_3} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathbf{G}_4} \Rightarrow 1]| \leq \frac{q_e(q_e - 1)}{2^{2n+1}}. \quad (13)$$

**G<sub>5</sub>**: This is the DETPRIV encryption oracle for  $\mathcal{E}^{\$}$ . We show the game **G<sub>4</sub>** is equivalent to the ideal world for  $\mathcal{E}^{\$}$  in **G<sub>5</sub>** by comparing the behaviors of both encryption oracles. By the assumption that the adversaries do not repeat queries, the query tuple  $(A, M)$  is always unique. By injective encoding algorithm  $\text{ENCODE}_{\tau}$ , the triple  $(A, M_L, M_R)$  is always unique as well. Let us focus on the encryption queries in the game **G<sub>4</sub>**. Therefore  $C_L$  is always uniformly random. As for the right output  $C_R$ , there are two possibilities. In the first case, tuple  $(C_L, M_R)$  hasn't been appeared before, and that makes  $\rho'(C_L, M_R)$  uniformly random. It follows that  $C_L$  be uniformly random. In the other case,  $C_L$  will be sampled uniformly at random. Thus, the output  $(C_L, C_R)$  will always be uniformly random, the same as it happens in the ideal world **G<sub>5</sub>**. So

$$|\Pr[\mathbf{A}^{\mathbf{G}_4} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathbf{G}_5} \Rightarrow 1]| = 0. \quad (14)$$

Finally, by combining the inequalities (9) to (14) we obtain the lemma bound.  $\square$

## B Proof of Lemma 2

Table 3. Games  $\mathbf{G}_0$ - $\mathbf{G}_1$ .

<b>initialize</b> <span style="float: right; border: 1px solid black; padding: 2px;"><math>\mathbf{G}_0</math></span>	<b>initialize</b> <span style="float: right; border: 1px solid black; padding: 2px;"><math>\mathbf{G}_1</math></span>
$K_1 \xleftarrow{\$} \mathcal{K}_1; K_2 \xleftarrow{\$} \mathcal{K}_2; K_3 \xleftarrow{\$} \mathcal{K}_3; K_4 \xleftarrow{\$} \mathcal{K}_4$ $\mathcal{X} \leftarrow \emptyset, \mathcal{Y} \leftarrow \emptyset$	$\tilde{\pi} \xleftarrow{\$} \widetilde{\text{Perm}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}}); K_4 \xleftarrow{\$} \mathcal{K}_4$ $\mathcal{X} \leftarrow \emptyset, \mathcal{Y} \leftarrow \emptyset, \mathcal{Z} \leftarrow \emptyset$
<b>oracle</b> $\text{Enc}(A, M)$ <b>if</b> $(A, M) \in \mathcal{X}$ <b>then</b> <b>return</b> $\perp$ $(M_L, M_R) \leftarrow \text{ENCODE}_{\tau}(M)$ $C_L \leftarrow E_{K_3}(M_L \oplus \mathcal{H}_{K_1 \  K_2}(A, M_R))$ $\oplus \mathcal{H}_{K_1 \  K_2}(A, M_R)$ $C_R \leftarrow \mathcal{E}_{K_4}(C_L, M_R)$ $\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}$ $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(A, C)\}$ <b>return</b> $(C_L \  C_R)$	<b>oracle</b> $\text{Enc}(A, M)$ <b>if</b> $(A, M) \in \mathcal{X}$ <b>then</b> <b>return</b> $\perp$ $(M_L, M_R) \leftarrow \text{ENCODE}_{\tau}(M)$ $C_L \leftarrow \tilde{\pi}((A, M_R), M_L)$ $C_R \leftarrow \mathcal{E}_{K_4}(C_L, M_R)$ $\mathcal{X} \leftarrow \mathcal{X} \cup \{(A, M)\}$ $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(A, C)\}$ <b>return</b> $(C_L \  C_R)$
<b>oracle</b> $\text{Dec}(A, C)$ <b>if</b> $(A, C) \in \mathcal{Y}$ <b>then</b> <b>return</b> $\perp$ $(C_L, C_R) \leftarrow C$ $M_R \leftarrow \mathcal{D}_{K_4}(C_L, C_R)$ $M_L \leftarrow E_{K_3}^{-1}(C_L \oplus \mathcal{H}_{K_1 \  K_2}(A, M_R))$ $\oplus \mathcal{H}_{K_1 \  K_2}(A, M_R)$ $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(A, C)\}$ <b>return</b> $\text{DECODE}_{\tau}(M_L, M_R)$	<b>oracle</b> $\text{Dec}(A, C)$ <b>if</b> $(A, C) \in \mathcal{Y}$ <b>then</b> <b>return</b> $\perp$ $(C_L, C_R) \leftarrow C$ $M_R \leftarrow \mathcal{D}_{K_4}(C_L, C_R)$ $M_L \leftarrow \tilde{\pi}^{-1}((A, M_R), C_L)$ $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(A, C)\}$ <b>return</b> $\text{DECODE}_{\tau}(M_L, M_R)$

*Proof.* We prove it by the game-playing technique. The games are in Table 3.

$\mathbf{G}_0$ : This is the DETAUTH encryption scheme for the RDCT construction.

$\mathbf{G}_1$ : In the game  $\mathbf{G}_1$ , we replace the tweakable blockcipher  $\tilde{E}$  with a tweakable random permutation  $\tilde{\pi} \xleftarrow{\$} \widetilde{\text{Perm}}(\mathcal{A} \times \mathcal{M}_{\mathcal{R}}, \mathcal{M}_{\mathcal{L}})$ . We bound  $\mathbf{A}$ 's distinguishing advantage between games  $\mathbf{G}_0$  and  $\mathbf{G}_1$  by constructing an STPRP adversary  $\mathbf{A}_1$  of  $\tilde{E}$  by [20]:

$$\begin{aligned}
 |\Pr[\mathbf{A}^{\mathbf{G}_0} \text{forge}] - \Pr[\mathbf{A}^{\mathbf{G}_1} \text{forge}]| &\leq \text{Adv}_{\tilde{E}, \tilde{E}^{-1}}^{\text{STPRP}}(q, \mathcal{O}(t)) \\
 &\leq \text{Adv}_{E, E^{-1}}^{\text{SPRP}}(q, \mathcal{O}(t+q)) + 3q^2\epsilon.
 \end{aligned} \tag{15}$$

It remains to bound the upper bound of  $\Pr [\mathbf{A}^{\mathbf{G}^1} \text{ forge}]$ . Let sets  $\mathcal{Z}, \mathcal{U}$  records all  $(A, C_L, C_R), ((A, M_R), C_L)$  respectively about **Enc** queries and **Dec** queries. Let  $*$  denote any bit string. For a new decryption query  $(A, C) \notin \mathcal{Y}$ , there are three possibilities about  $(A, C_L, C_R)$ .

1.  $(A, *, *)$  has not been appeared in  $\mathcal{Z}$ . Then  $((A, M_R), C_L)$  has not been appeared in  $\mathcal{U}$ ;
2.  $(*, C_L, *)$  has not been appeared in  $\mathcal{Z}$ . Then  $((A, M_R), C_L)$  has not been appeared in  $\mathcal{U}$ ;
3.  $(*, *, C_R)$  has not been appeared in  $\mathcal{Z}$ . There are two possibilities. In the first case,  $(*, C_L, *)$  has not been appeared in  $\mathcal{Z}$ . Then  $((A, M_R), C_L)$  has not been appeared in  $\mathcal{U}$ ; In the other case,  $(*, C_L, *)$  have been appeared in  $\mathcal{Z}$ . We assume it is  $(*, C_L, C'_R)$  where  $C'_R \neq C_R$ . Then  $M'_R \neq M_R$ . So  $((A, M_R), C_L)$  has not been appeared in  $\mathcal{U}$  as well.

Therefore, the inputs of  $\tilde{\pi}^{-1}$ :  $((A, M_R), C_L)$  always don't repeat. By the STPRP security of  $\tilde{\pi}$ ,

$$\begin{aligned} \Pr [\mathbf{A}^{\mathbf{G}^1} \text{ forge}] &\leq \frac{2^{2n-\tau}}{2^{2n}-q_e} + \frac{2^{2n-\tau}}{2^{2n}-q_e-1} + \cdots + \frac{2^{2n-\tau}}{2^{2n}-q_e-q_d+1} \\ &\leq \frac{2^{2n-\tau} q_d}{2^{2n}-q} \leq \frac{q}{2^\tau - \frac{q}{2^{2n-\tau}}} \leq \frac{q}{2^\tau - q}. \end{aligned} \quad (16)$$

Finally, by combining the inequalities (15) to (16) we obtain the lemma bound.  $\square$