Double Ciphertext Mode : A Proposal for Secure Backup

Debrup Chakraborty and Cuauhtemoc Mancillas-López

Computer Science Department
Centro de Investigaciones y Estudios Avanzados del IPN
Av. IPN 2508, San Pedro Zacatenco
Mexico D.F. 07360, Mexico

Abstract. Security of data stored in bulk storage devices like the hard disk has gained a lot of importance in the current days. Among the variety of paradigms which are available for disk encryption, low level disk encryption is well accepted because of the high security guarantees it provides. In this paper we view the problem of disk encryption from a different direction. We explore the possibility of how one can maintain secure backups of the data, such that loss of a physical device will mean neither loss of the data nor the fact that the data gets revealed to the adversary. We propose an efficient solution to this problem through a new cryptographic scheme which we call as the double ciphertext mode (DCM). In this paper we describe the syntax of DCM, define security for it and give some efficient constructions. Moreover we argue regarding the suitability of DCM for the secure backup application and also explore other application areas where a DCM can be useful.

1 Introduction

It has been estimated that the laptop loss rates are around 2% per year [4], which signifies that an organization with 100,000 laptops, loses on average several of them per day. A lost laptop amounts to two severe consequences: (1) the loss of the data stored in it (2) the stored data getting revealed to an adversary. The later problem has been identified to be more severe than the former one, as it is rightly argued that if sensitive data gets into the hands of an unwanted person he/she can potentially cause much greater damage than the one incurred by the mere loss of the data. As a solution to problem (2) a variety of disk encryption schemes have been proposed in the last few years. One of the well accepted proposals for disk encryption is *low level disk encryption.* In low level disk encryption, the encryption algorithm resides in the disk controller, it views the disk as a collection of disk sectors and has no knowledge of the high level partitions of the disk like files and directories. It encrypts a sector before writing it and decrypts a sector after reading it and before sending to the operating system. In [6] it was pointed out that low level disk encryption can be achieved by a cryptographic scheme called the tweakable enciphering scheme. Tweakable enciphering schemes are a special type of block-cipher mode of operation which are length preserving and provides security in the sense of a strong pseudo random permutation. In the last few years there has been a lot of activities towards developing secure and efficient tweakable enciphering schemes [2, 3, 5–7, 13, 14, 16].

Problem (1) has not been adequately addressed in the literature. A trivial solution of problem (1) is to maintain backups of the data securely. With the advent of cheap storages and backup technologies which allows synchronous read/write operations even from remote storage devices keeping backup is easy. So one may think that the data always gets written to two storage devices both equipped with a low level disk encryption algorithm (as provided by tweakable enciphering schemes), thus loss of one device does not result in any of the consequences stated before. In this paper we handle the problem of backup but in a bit different way than the trivial solution. In the trivial solution, as stated, each plain text will have associated with it two cipher texts (possibly different as one can use two different algorithms or two different keys for encryption in the two different devices), and in a normal scenario both these ciphertexts would be available to the user. We suggest a scheme which can exploit this redundancy and thus give rise to more efficient ways to solve this problem.

Our goal is to design a symmetric key encryption scheme which produces two ciphertexts C^{L} and C^{R} (L and R can be read as local and remote) and a tag τ for a given plaintext P with the action of one or more secret keys. The requirements on C^{L} and C^{R} would be the following:

1. C^{L}, C^{R} and P must be of same length.
2. There should exist a very efficient function g such that $g(C^{\mathsf{L}}, C^{\mathsf{R}}) = P$ and g should not require any secret parameter.
3. One must be able to recover P from either C^{L} or C^{R} by using the secret key and the tag.

The rationale behind such a goal is as follows. For the secure backup scenario, we assume plaintexts to be disk sectors and the corresponding ciphers would also get written in disk sectors, hence the two ciphertexts must be of the same length of that of the plaintext. When an user have access to both C^{L} and C^{R} then (s)he can very efficiently produce the plain text which produced them using the function g. If one of C^{L} or C^{R} is lost then with the knowledge of the secret key and the tag, P can be recovered from one of C^{L} or C^{R}. The function g should be such that obtaining P through it should be much more efficient than obtaining P by decryption of either C^{L} or C^{R}.

Additionally, we require the scheme to be secure in some sense. For arguing about security, given plaintext P, we rule out the possibility of adversarial access to both C^{L} and C^{R}. This is not very un-natural as we may assume that C^{L} is stored in the laptop of an user where as C^{R} gets stored in a trusted location provided by his/her employer. It may be difficult for an adversary to have access to both versions of the ciphertext. The security goal for the scheme is that, it should be difficult for any computationally bounded adversary to distinguish between the outputs of the encryption scheme from random strings when the adversary can see either C^{L} or C^{R} along with the tag for messages of his choice. Additionally by looking at either C^{L} or C^{R} along with the tag for messages of his/her choice, (s)he should be unable to produce a new ciphertext tag pair which would get decrypted.

Our Contribution: We analyze the above stated problem and propose a solution to it. Our solution is a special encryption algorithm which we call the *double ciphertext mode* (DCM). We describe the syntax of a DCM and define security for it. We provide a generic construction for a DCM using a pseudorandom function and a block-cipher, and also a specific construction which requires only one block-cipher key. We prove security for both the constructions. We observe the similarity of DCM constructions with a specific class of deterministic authenticated encryption schemes [12]. Finally we mention other application areas where we think a DCM can be useful.

2 Preliminaries and Notations

We denote the set of all binary strings as $\{0,1\}^*$. For $X, Y \in \{0,1\}^*$, $X||Y$ will denote the concatenation of the strings X and Y. For a string X, $\mathsf{bin}_n(X)$ will denote the n bit binary representation of X and $\mathsf{len}(X)$ will denote the length of X in bits. Let $\mathsf{ty} \in \{\mathsf{L}, \mathsf{R}\}$, if $\mathsf{ty} = \mathsf{L}$ then $\bar{\mathsf{ty}} = \mathsf{R}$, and vice versa. By $X \xleftarrow{\$} \mathcal{S}$, we will denote the event of choosing X uniformly at random from the finite set $\mathcal{S}$.

We shall sometimes view n bit strings as polynomials of degree less than n with coefficients in $GF(2)$, thus we would view n bit strings as elements of $GF(2^n)$. For n-bit strings X, Y, $X \oplus Y$ will denote the addition in the field, which is a bitwise xor of the strings, and XY will denote multiplication in the field which can be realized by ordinary polynomial multiplication modulo the irreducible polynomial representing the field. The monomial x is an element of $GF(2^n)$ which can be represented by the n bit binary representation of the integer 2. Given $X \in GF(2^n)$, by xX we shall mean the multiplication of polynomial X with the monomial $x \in GF(2^n)$. Similarly $(1 \oplus x)X$ will mean the multiplication of $(1 \oplus x) \in GF(2^n)$ with X in the field. The operation xX can be efficiently realized with a shift and conditional xor. The operation $(1 \oplus x)X$ would require an additional xor.

An adversary $\mathcal{A}$ is an algorithm which has access to one or more oracles and which outputs either 0 or 1. Oracles are written as superscripts. The notation $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2} \Rightarrow 1$ denotes the event that the adversary $\mathcal{A}$, interacts with the oracles $\mathcal{O}_1, \mathcal{O}_2$, and finally outputs the bit 1.

A pseudo-random function F with domain $\mathcal{X}$ and range $\mathcal{Y}$ is a map $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ where $\mathcal{K}$ is the key-space. We generally write $F_K(.)$ instead of $F(K, .)$. Let $\mathrm{Func}(\mathcal{X}, \mathcal{Y})$ denote the all possible functions from $\mathcal{X}$ to $\mathcal{Y}$. If $\mathcal{X} = \{0,1\}^n$ and $\mathcal{Y} = \{0,1\}^m$ then we denote $\mathrm{Func}(\mathcal{X}, \mathcal{Y})$ as $\mathrm{Func}(n, m)$.

The PRF advantage of an adversary $\mathcal{A}$ in distinguishing $F(.,.)$ from a random function is defined as

$$\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) = \Pr\left[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{F(K,.)} \Rightarrow 1\right] - \Pr\left[\rho \xleftarrow{\$} \mathrm{Func}(\mathcal{X}, \mathcal{Y}) : \mathcal{A}^{\rho(.)} \Rightarrow 1\right].$$

For pseudorandom functions another adversarial goal is useful which is known as forgery. For defining forgery for a pseudorandom function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, an adversary $\mathcal{A}$ is given oracle access to $F_K(.)$, where $K \xleftarrow{\$} \mathcal{K}$. Suppose $\mathcal{A}$ makes q queries X_i, $1 \leq i \leq q$ to its oracle and gets back $Y_i = F_K(X_i)$, $1 \leq i \leq q$. At the end $\mathcal{A}$ attempts a forgery by producing a pair $(\tilde{X}, t)$ such that $\mathcal{A}$ never obtained t as a response to any of his previous queries X_i $(1 \leq i \leq q)$. $\mathcal{A}$ is said to be successful if $F_K(\tilde{X}) = t$. It is well known that (for example see [15])

$$\Pr[\mathcal{A}^{F_K(\cdot)} \text{ forges}] \leq \mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) + \frac{1}{|\mathcal{Y}|} \tag{1}$$

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 $E(K,.)$. Let $\mathrm{Perm}(n)$ denote the set of all permutations on $\{0,1\}^n$. For $K \xleftarrow{\$} \mathcal{K}$, we require $E_K(.)$ to be a pseudorandom permutation. The advantage of an adversary $\mathcal{A}$ in breaking the pseudorandomness of $E(,)$ is defined in the following manner.

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

3 Our Proposal: Double Ciphertext Mode (DCM)

We fix m, n, ℓ as positive integers. Let, $\mathbb{M} = \{0,1\}^{nm}$, $\mathbb{C} = \{0,1\}^{nm+\ell}$, $\mathbb{T} = \{0,1\}^n$ be the message space and cipher space and tweak space respectively. A double ciphertext mode (DCM) is a set of four algorithms $(\mathbb{K}, \mathcal{E}, \mathcal{D}, g)$, where $\mathbb{K}$ is the key generation algorithm (we shall also denote the key space with $\mathbb{K}$), $\mathcal{E}$ the encryption algorithm, $\mathcal{D}$ the decryption algorithm, and g a special function which we call the *recovery function.*

The encryption function $\mathcal{E} : \mathbb{K} \times \mathbb{T} \times \mathbb{M} \times \{\mathsf{R}, \mathsf{L}\} \to \mathbb{C}$ takes in a key, a tweak, a message and type (which can be either R or L)to produce a cipher. For any given $M \in \mathbb{M}$, $K \in \mathbb{K}$, $T \in \mathbb{T}$, and $\mathsf{ty} \in \{\mathsf{R}, \mathsf{L}\}$ we shall write $\mathcal{E}(K, T, M, \mathsf{ty})$ as $\mathcal{E}_K(T, M, \mathsf{ty})$ or sometimes as $\mathcal{E}_K^{T,\mathsf{ty}}(M)$. Any cipher $\mathcal{C}$ produced by the encryption function from a message M can be parsed as $\mathcal{C} = C||\tau$, where $|C| = |M|$ and $|\tau| = \ell$. We define functions cipher and tag, such that given $\mathcal{C} = C||\tau \in \mathbb{C}$, $\mathsf{cipher}(\mathcal{C}) = C$ and $\mathsf{tag}(\mathcal{C}) = \tau$. Also, for any $M \in \mathbb{M}$, $K \in \mathbb{K}$, $T \in \mathbb{T}$, $\mathsf{tag}(\mathcal{E}_K^{T,\mathsf{L}}(M)) = \mathsf{tag}(\mathcal{E}_K^{T,\mathsf{R}}(M))$.

$\mathcal{D} : \mathbb{K} \times \mathbb{T} \times \mathbb{C} \times \{R, L\} \to \mathbb{M} \cup \{\perp\}$ is the decryption algorithm, $\mathcal{D}(K, T, \mathcal{C}, \mathsf{ty})$ is denoted as $\mathcal{D}_K(T, \mathcal{C}, \mathsf{ty})$ or sometimes as $\mathcal{D}_K^{T,\mathsf{ty}}(M)$. And,

$$\begin{aligned} \mathcal{D}_K(T, \mathcal{C}, \mathsf{ty}) &= M, \text{ if } \mathcal{C} = \mathcal{E}_K(T, M, \mathsf{ty}), \\ &= \perp, \text{ otherwise.} \end{aligned}$$

The function g is a public function which does not use any key. Let $\mathcal{C}^{\mathsf{L}} = \mathcal{E}_K(T, M, \mathsf{L})$ and $\mathcal{C}^{\mathsf{R}} = \mathcal{E}_K(T, M, \mathsf{R})$, then $g(\mathsf{cipher}(\mathcal{C}^{\mathsf{L}}), \mathsf{cipher}(\mathcal{C}^{\mathsf{R}})) = M$. The existence of this special recovery function g makes a DCM scheme different from other encryption schemes. This function g enables decryption of a ciphertext given the two versions of it (i.e., both the L and R versions) without the key. For a DCM to be practically useful, it is required that the recovery function g can be computed much more efficiently than the decryption function $\mathcal{D}$.

We define DCM with fixed values of m, n and ℓ, where n can be seen as the block-length of the block-cipher. Thus, the message length is fixed to some multiple of n and the tweak length is always n. We define in this manner keeping in mind the specific application, where a more general definition is not required. But a DCM can be defined more generally where the message length, tweak length and tag length are not fixed.

4 Secure Backup Through DCM

The above syntax of a DCM has been constructed keeping in mind the specific need of a secure backup scheme. In what follows we try to describe how a DCM can be used for keeping secure backup using an example. This example is meant to be motivational, we do not try to detail technical concerns for a true implementation of a DCM.

A user (U) works with sensitive data in his office which he stores in his laptop. U cannot afford to lose the data or to reveal the data to an adversary. The employer of U has a server (S), where U can keep backups. Let us consider that U mirrors the disk of his laptop with that of the server S. The disk controllers of the laptop and the server are both are equipped with a DCM instantiated with the same keys. The type ty is set to L and R in the laptop and the server respectively, i.e., in the laptop the encryption algorithm $\mathcal{E}_K(.,.,\mathsf{L})$ runs and in the server the encryption algorithm $\mathcal{E}_K(.,.,\mathsf{R})$ runs. When a plaintext M is to be written in the sector with address T, then $\mathsf{cipher}(\mathcal{E}_K(T, M, \mathsf{L}))$ and $\mathsf{cipher}(\mathcal{E}_K(T, M, \mathsf{R}))$ gets written in the sector with address T of the laptop and the server respectively. The tag produced by the encryption algorithms in both the server and the laptop would be the same and one of it have to be stored. Without loss of generality we consider that in the server, the $\mathsf{tag}(\mathcal{E}_K(T, M, \mathsf{R}))$ is communicated to the operating system which stores it in another disk indexed with the address T. Note that the tag produced for the ciphertext stored in address T is not stored in the sector, but it is stored in a different location. The operating system maintains the tags with the related disk sector addresses through a software solution. Note, that the tags can be public as this is a part of the ciphertext and need not be stored securely.

Now let us discuss about decryption. In the normal scenario, when nothing unnatural has happened to the laptop of U, then to recover the contents of sector T, U requests a read from both the disks of his laptop and the server to get $\mathsf{cipher}(\mathcal{E}_K(T, M, \mathsf{L}))$ and $\mathsf{cipher}(\mathcal{E}_K(T, M, \mathsf{R}))$, and recovers M by the recovery function g. In an un-natural scenario when data in one of the disks is lost then U applies the decryption function on the disk sector along with the tag to get back the plaintext.

When a plaintext is recovered using the function g then the tag is not used, so the integrity of the message is not checked. But U can check for the integrity of the messages in the disk sectors at any point he want. He can schedule regular integrity checks similar to virus scans even if he believes that the scenario is normal. This implies that U need not trust the server in which his backup is being kept.

5 Security of DCM

We want to make DCM secure in two ways, in terms of privacy and authentication. Firstly, as stated earlier, we assume that an adversary attacking DCM should have access to only one of the versions of the ciphertext, i.e., either the L or the R version. We give the adversary the liberty to see one version (either L or R) of the ciphertext along with the tag for plaintexts of his choice and we say that the adversary breaks DCM in the sense of privacy if the adversary can distinguish between the obtained ciphertexts and random strings. This is the usual notion of privacy as applicable to symmetric key encryption schemes but here the adversary is restricted in the sense that it can only view one version of the ciphertext. The other form of security that we want is that of authentication, that is an adversary must not be able to modify ciphertexts such that the modified ones gets decrypted, i.e., it should be difficult for an adversary to create a new version of ciphertext and tag pair which will get decrypted even if he has seen ciphertexts tag pairs of plaintexts of his choice. We formalize these notions of security next.

Let $\Psi = (\mathbb{K}, \mathcal{E}, \mathcal{D}, g)$ be a DCM scheme. Let $\mathcal{A}$ be an adversary attacking Ψ. The adversary has access to the oracle $\mathcal{O}$. $\mathcal{O}$ can either be $\mathcal{E}_K(.,.,.)$, where K is generated uniformly at random from $\mathbb{K}$ or it can be $\$(.,.,.)$ which returns random strings of length equal to the length of the ciphertext. The adversary can query the oracle with a query of the form (T, M, ty), where M, T denotes the message and tweak respectively and $\mathsf{ty} \in \{\mathsf{L}, \mathsf{R}\}$, and thus get back the responses from the oracle. $\mathcal{A}$ has the following query restrictions:

1. $\mathcal{A}$ cannot query the encryption oracle with the same message and tweak on both values of ty, i.e., $\mathcal{A}$ is not allowed to see both the L and R versions of the ciphertext, for the same message tweak pair.
2. The adversary does not repeat a query.

We define the privacy advantage of adversary $\mathcal{A}$ as follows:

$$\mathbf{Adv}_{\Psi}^{\text{dcm-priv}}(\mathcal{A}) = \Pr\left[K \xleftarrow{\$} \mathbb{K} : \mathcal{A}^{\mathcal{E}_K(.,.,.)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(.,.,.)} \Rightarrow 1\right].$$

To define the authentication advantage we consider that $\mathcal{A}$ is given access to the encryption oracle $\mathcal{E}_K(.,.,.)$ and queries it with plaintexts of his choice and obtains the corresponding ciphertexts. Finally, $\mathcal{A}$ outputs a forgery, which consists of a tweak, a ciphertext and a type $(T, \mathcal{C}, \mathsf{ty})$. $\mathcal{A}$ is said to be successful if $\mathcal{D}_K(T, \mathcal{C}, \mathsf{ty}) \neq \perp$. For querying the encryption oracle $\mathcal{A}$ follows the same restrictions as described above additionally $\mathcal{A}$ have the following restrictions:

1. $\mathcal{A}$ cannot output $(T, \mathcal{C}, \mathsf{ty})$ as a forgery if he obtained $\mathcal{C}$ as a response for his query (T, X, ty) to the encryption oracle.
2. $\mathcal{A}$ cannot output $(T, \mathcal{C}, \mathsf{ty})$ as a forgery if $g(\mathsf{cipher}(\mathcal{C}), \mathsf{cipher}(\tilde{\mathcal{C}})) = X$ where $\tilde{\mathcal{C}}$ was obtained as a response to his encryption query $(T, X, \bar{\mathsf{ty}})$.

Both restrictions are to rule out trivial wins. The second restriction is particularly interesting, note that if $\tilde{C}||\tau = \mathcal{E}_K(T, X, \mathsf{R})$ then it may be easy for an adversary to guess (or compute) C, where $C||\tau = \mathcal{E}_K(T, X, \mathsf{L})$ without querying the encryption oracle, as g is a public function and $g(C, \tilde{C}) = X$. Thus $\mathcal{A}$ may trivially produce the forgery $(T, C||\tau, \mathsf{L})$ knowing that $\tilde{C}||\tau = \mathcal{E}_K(T, X, \mathsf{R})$ and $g(C, \tilde{C}) = X$.

The authentication advantage of the adversary $\mathcal{A}$ is defined as the probability that $\mathcal{A}$ does a successful forgery, in other words

$$\mathbf{Adv}_{\Psi}^{\text{dcm-auth}}(\mathcal{A}) = \Pr[\mathcal{A}^{\mathcal{E}_K(.,.,.)} \text{ forges }]$$

We consider Ψ to be secure, if both $\mathbf{Adv}_{\Psi}^{\text{dcm-priv}}(\mathcal{A})$ and $\mathbf{Adv}_{\Psi}^{\text{dcm-auth}}(\mathcal{A})$ are small for every computationally bounded adversary $\mathcal{A}$.

6 DCMG: A generic construction of DCM

We construct an DCM scheme using two basic primitives a pseudo-random function and a blockcipher secure in the sense of a pseudo-random permutation. We call our construction as DCMG.

Let $F : \mathcal{K}_1 \times \{0,1\}^{mn+n} \to \{0,1\}^n$ be a pseudo-random function and $E : \mathcal{K}_2 \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher secure in the sense of a pseudo-random permutation. We construct a DCM scheme using F and E which we call as DCMG$[F, E]$. The message space of DCMG is $\{0,1\}^{mn}$ and the tweak space is $\{0,1\}^n$. We describe the encryption and decryption algorithms using DCMG in Fig. 1.

The algorithms in Fig. 1 are self explanatory. Using the pseudo-random function F a tag τ is produced which acts as the initialization vector of a counter type mode of operation. The counter mode produces blocks of random strings, which are mixed with the plaintext. For producing ciphertexts of two different types the output of the counter mode are mixed in two different way. The specific way in which this mixing is done is to enable the existence of an efficient function g. In this case the function g is simple, we define g as $g(X, Y) = X \oplus Y$. Let $C^{\mathsf{L}}||\tau = \mathcal{E}_{K_1,K_2}^{T,\mathsf{L}}(P_1, \ldots, P_m)$ and $C^{\mathsf{R}}||\tau = \mathcal{E}_{K_1,K_2}^{T,\mathsf{R}}(P_1, \ldots, P_m)$, It is easy to verify that $C^L \oplus C^R = P_1|| \ldots ||P_m$.

Fig. 1. Encryption and decryption using DCMG$[F, E]$. K_1 is the key for the PRF and K_2 the block-cipher key. $\mathsf{len}(P_i) = \mathsf{len}(C_i) = n$, $1 \le i \le m$.

```
Algorithm DCMG.$\mathcal{E}^{T,\mathsf{ty}}_{K_1,K_2}(P_1, \ldots, P_m)$

1. $\tau \leftarrow F_{K_1}(P_1||P_2||\ldots||P_m||T)$;
2. for $j = 1$ to $m$ ;
3.     $R_j \leftarrow E_{K_2}(\tau \oplus \mathsf{bin}_n(j))$;
4.     if $\mathsf{ty} = \mathsf{L}$
5.         $C_j \leftarrow R_j \oplus (1 \oplus x)P_j$
6.     else $C_j \leftarrow R_j \oplus xP_j$
7.     endif
8. endfor
9. return $(C_1||C_2||\ldots||C_m||\tau)$;
```

```
Algorithm DCMG.$\mathcal{D}^{T,\mathsf{ty}}_{K_1,K_2}(C_1, \ldots, C_m, \tau)$

1. for $j = 1$ to $m$ ;
2.     $R_j \leftarrow E_{K_2}(\tau \oplus \mathsf{bin}_n(j))$;
3.     if $\mathsf{ty} = \mathsf{L}$
4.         $P_j \leftarrow (C_j \oplus R_j)(1 \oplus x)^{-1}$
5.     else $P_j \leftarrow (C_j \oplus R_i)x^{-1}$
6.     endif
7. endfor
8. $\tau' \leftarrow F_{K_1}(P_1||P_2||\ldots||P_m||T)$;
9. if $\tau' = \tau$ return $(P_1, \ldots, P_m)$ else return $\perp$;
```

7 Characteristics of the construction

1. Efficiency: The construction requires a pseudo-random function (PRF) which can be replaced by a secure keyed message authentication code. Other than the call to the PRF, the scheme requires m block-cipher calls to encrypt a m block message. Decryption is costlier than encryption as decryption requires all the operations necessary for encryption plus a multiplication by x^{-1} or $(1+x)^{-1}$ per block of message. Thus, in total decryption requires m block cipher calls and m finite field multiplications in addition to the call to the PRF. The inverses which are required to be computed for decryption can be pre-computed, thus explicit computation of inverses would not be required for decryption. Also note that the multiplications required can be performed quite efficiently using pre-computation as one of the operands is fixed, and this operand (either x^{-1} or $(1 \oplus x)^{-1}$) does not contain any key related materials. As required, the function g can be computed much more efficiently than the decryption function, as for recovering a m block message with g, one needs to perform only m xors of n bit strings.
2. Number of keys: This generic construction requires more than one key, one for the block-cipher and another (or more) for the PRF depending on the number of keys required by the PRF. This requirement can be reduced, later in the paper we describe a particular construction which requires only one block-cipher key.
3. Message length restrictions: The construction only works for fixed length messages which are multiples of the block length of the block-cipher. We do not know any trivial way to remove this restriction from DCMG, as if there is a broken block (a block of length less than the block length of the block-cipher)then there could be a ciphertext expansion (see lines 5 and 6 of the encryption algorithm in Fig. 1). But this restriction is not severe keeping in mind the application, as it is assumed that the two ciphertexts produced would be disk sectors to be stored in two different disk volumes.
4. The tweak length: The construction is specified for fixed length tweaks, where the tweak length is the same as the block length of the underlying blockcipher. For the application of the disk sector encryption, the sector address is considered as the tweak which is likely to be of fixed length.

8 Security of DCMG

The following theorems suggests the security of DCMG

Theorem 1. *Let $\Upsilon = \mathrm{Func}(mn+n,n)$ and $\Pi = \mathrm{Perm}(n)$. Let $\mathcal{A}$ be an adversary attacking* $\mathrm{DCMG}[\Upsilon,\Pi]$ *who asks q queries, then*

$$\mathbf{Adv}^{\text{dcm-priv}}_{\mathrm{DCMG}[\Upsilon,\Pi]}(\mathcal{A}) \leq \frac{m^2q^2}{2^n} \tag{2}$$

$$\mathbf{Adv}^{\text{dcm-auth}}_{\mathrm{DCMG}[\Upsilon,\Pi]}(\mathcal{A}) \leq \frac{1}{2^n} \tag{3}$$

Theorem 2. *Let $F : \mathcal{K}_1 \times \{0,1\}^{mn+n} \to \{0,1\}^n$ be a PRF and $E : \mathcal{K}_2 \times \{0,1\}^n \to \{0,1\}^n$ be a block-cipher secure in the PRP sense. Let $\mathcal{A}$ be an adversary attacking* $\mathrm{DCMG}[F,E]$ *who asks q queries, then their exists adversaries $\mathcal{A}'$ and $\mathcal{A}''$ such that*

$$\mathbf{Adv}^{\text{dcm-priv}}_{\mathrm{DCMG}[F,E]}(\mathcal{A}) \leq \frac{m^2q^2}{2^n} + \mathbf{Adv}^{\text{prf}}_{F}(\mathcal{A}') + \mathbf{Adv}^{\text{prp}}_{E}(\mathcal{A}'') \tag{4}$$

$$\mathbf{Adv}^{\text{dcm-auth}}_{\mathrm{DCMG}[F,E]}(\mathcal{A}) \leq \frac{1}{2^n} + \frac{m^2q^2}{2^n} + \mathbf{Adv}^{\text{prf}}_{F}(\mathcal{A}') + \mathbf{Adv}^{\text{prp}}_{E}(\mathcal{A}'') \tag{5}$$

where $\mathcal{A}'$ and A'' asks $O(q)$ queries and run for time $t + O(q)$ where t is the running time of $\mathcal{A}$.

Theorem 1 depicts the information theoretic bound where as Theorem 2 shows the complexity theoretic bound. Transition from Theorem 1 to 2 is quite standard and we shall not present it in this paper. We provide the full proof of Theorem 1 in Appendix A. But here we shall provide some motivation regarding why Theorem 1 is true.

In the construction which is being referred to in Theorem 1 the pseudorandom function F is replaced by a function ρ chosen uniformly at random from $\Upsilon = \mathrm{Func}(mn+n,n)$ and the blockcipher E is replaced by a permutation π chosen uniformly at random from $\Pi = \mathrm{Perm}(n)$. To prove the privacy bound note that the inputs to the function ρ would be all new, as the adversary is not allowed to repeat a query. Also, as ρ is a random function so the tag (which is the output of ρ) would be random. The only way the adversary can distinguish the output of $\mathrm{DCMG}[\Upsilon,\Pi]$ from random strings if there is a collision in the domain or the range sets of the permutation π, the probability of this collision in $m^2q^2/2^n$, which suggests the bound in eq. (2). For proving authenticity we give access of the random permutation π to the adversary, as this permutation π is independent of the random function ρ the adversary have no added advantage of forgery. With access to π the adversary will know what would be the input to the random function ρ for the forgery he produces. Given his query restrictions, the probability that we can produce a successful forgery would be less than $1/2^n$ (for details see the full proof in Appendix A).

9 Similarity with Deterministic Authenticated Encryption

Deterministic authenticated encryption (DAE) schemes are deterministic encryption schemes which provides security both in terms of privacy and authentication. These schemes were first proposed by Rogaway and Shrimpton [12], where the authors described the syntax of the DAE along with the security requirements and a generic construction which they called as the Synthetic Initialization Vector (SIV) mode. The SIV mode construction uses a secure IV based privacy only encryption mode like the counter mode or the CBC mode along with a special type of pseudorandom function which takes as input a vector of binary strings. Let $\{0,1\}^{**}$ denote the space for all vectors of binary strings, and if X and Y are vectors such that $X = (X_1,\ldots,X_m)$ and $Y = (Y_1,\ldots,Y_{m'})$, then $[[X,Y]] = (X_1,\ldots,X_m,Y_1,\ldots,Y_{m'})$. The SIV mode $\mathsf{SIV}[\boldsymbol{F},\mathsf{Priv}]$ is constructed using a pseudorandom function $\boldsymbol{F} : \mathcal{K}_1 \times \{0,1\}^{**} \to \{0,1\}^n$ and a privacy only encryption scheme $\mathsf{Priv} = (\mathcal{K}_2, \boldsymbol{E}, \boldsymbol{D})$ where $\mathcal{K}_2$ is the key space, $\boldsymbol{E} : \mathcal{K}_2 \times \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^*$ is the encryption algorithm and $\boldsymbol{D} : \mathcal{K}_2 \times \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^*$ is the decryption algorithm. For an initialization vector IV, and key K the encryption and decryption algorithms are written as $\boldsymbol{E}^{IV}_K(.)$ and $\boldsymbol{D}^{IV}_K(.)$ respectively. The $\mathsf{SIV}[\boldsymbol{F},\mathsf{Priv}]$ construction is shown in Fig. 2.

Fig. 2. Encryption and decryption using $\mathsf{SIV}[\boldsymbol{F}, \mathsf{Priv}]$. K_1 is the key for the PRF and K_2 the key for the privacy only encryption scheme $\mathbf{E}$. The header T is a vector of binary strings.

Algorithm $\mathsf{SIV}.\mathrm{Encrypt}^{T}_{K_1,K_2}(P)$	**Algorithm** $\mathsf{SIV}.\mathrm{Decrypt}^{T}_{K_1,K_2}(C,\tau)$
1. $\tau \leftarrow \boldsymbol{F}_{K_1}([[P,T]])$;	1. $P = \boldsymbol{D}^{\tau}_{K_2}(C)$;
2. $C \leftarrow \boldsymbol{E}^{\tau}_{K_2}(P)$;	2. $\tau' = \boldsymbol{F}_{K_1}([[P,T]])$;
3. **return** $C\|\tau$;	4. **if** $\tau' = \tau$ **return** P else **return** $\perp$;

One can note the similarity between the generic construction of a DCM with that of the SIV scheme. A SIV scheme which uses the counter-mode for the secure privacy only scheme can be suitably re-worked to make a DCM following the construction in Section 6. The stream of random bits which would be generated by the counter mode needs to be xor-ed with the plaintext in two different ways to get the two different versions of the ciphertext. But in the generic DAE scheme as described in [12] any privacy only scheme can be used and also have the provision of vector headers (headers are equivalent to the tweak), so we do not use the syntax of a DAE to define DCM, but we just note the similarity.

The security of DAE as described in [12] is also similar to that of a DCM, but a DAE adversary does not query a type to its oracle, as there is no such possibility in case of a DAE. In [12] a combined advantage for both security and authentication was proposed, which is elegant, but we feel that the proofs are still simpler if the privacy and authenticity advantages are stated separately, so we define the privacy and authentication advantages separately.

10 Constructing a DCM Using BRW Polynomials

Here we provide a particular construction of a DCM, where we design the pseudorandom function using a special type of polynomials called the BRW polynomials. Also our construction requires only one blockcipher key. We call the construction as DCM-BRW.

10.1 Bernstein-Rabin-Winograd Polynomials

In [1], Bernstein builds upon a previous work by Rabin and Winnograd [11] to define a sequence of special type of polynomials which are called as the BRW polynomials in [14]. We define BRW polynomials over $GF(2^n)$ as that is what would be useful for us. Let $X_1, X_2, \ldots, X_m \in GF(2^n)$, and also let $h \in GF(2^n)$ then BRW polynomials are defined as follows

- $\mathsf{BRW}_h() = 0$
- $\mathsf{BRW}_h(X_1) = X_1$
- $\mathsf{BRW}_h(X_1, X_2) = X_1 h \oplus X_2$
- $\mathsf{BRW}_h(X_1, X_2, X_3) = (h \oplus X_1)(h^2 \oplus X_2) \oplus X_3$
- $\mathsf{BRW}_h(X_1, X_2, \ldots, X_m) = \mathsf{BRW}_h(X_1, \ldots, X_{t-1})(h^t \oplus X_t) \oplus \mathsf{BRW}_h(X_{t+1}, \ldots, X_m)$, if $t \in \{4, 8, 16, 32, \ldots\}$ and $t \leq m < 2t$.

Next we state some of the important and useful properties of BRW polynomials without proof. The interested reader is refered to [1, 11, 14]

Property 1. If $m \geq 3$ and $t \leq m < 2t - 1$ then $\mathsf{BRW}_h(X_1, \ldots, X_m)$ is a monic polynomial of degree $2t - 1$.

Property 2. For $m \geq 2$, $\mathsf{BRW}_h(X_1, \ldots, X_m)$ can be computed using $\lfloor m/2 \rfloor$ multiplications and $\lg m$ squarings.

Property 3. Let $X = (X_1, X_2, \ldots, X_m), X' = (X'_1, X'_2, \ldots, X'_m) \in [GF(2^n)]^m$ such that $X \neq X'$. Let $Y = \mathsf{BRW}_h(X_1, X_2, \ldots, X_m)$ and $Y' = \mathsf{BRW}_h(X'_1, X'_2, \ldots, X'_m)$. Then for every fixed $a \in GF(2^n)$

$$\Pr[Y \oplus Y' = a] = \frac{(2m-1)}{2^n}$$

The probability is taken over the random choice of h.

For the construction that we present next property 2 would be important to see the efficiency of out construction and property 3 would be important to prove the security of the construction.

10.2 The Construction

The generic construction DCMG that we proposed in Section 6 uses a pseudorandom function along with a block-cipher in a counter type mode of operation. In the specific construction DCM-BRW we replace the general pseudorandom function with a specific pseudorandom function which uses a BRW polynomial. The encryption and decryption algorithms using DCM-BRW are shown in Fig. 3. The construction requires two keys, a key h for the BRW polynomial and a key K for the block-cipher. Other than the keys the encryption algorithm takes in the plaintext, the tweak and the type and returns a ciphertext and the decryption algorithm takes in the ciphertext, tweak and type and returns the plaintext. The details of the working of the algorithm are self explanatory as depicted in Fig. 3. For this construction it is required that the irreducible polynomial representing the field $GF(2^n)$ is primitive.

Fig. 3. Encryption and decryption using DCM-BRW.

```
Algorithm DCM-BRW.$\mathcal{E}_{h,K}^{T,\mathsf{ty}}(P_1, \ldots, P_m)$

1.  $\alpha = E_K(0); \beta = E_K(1);$
2.  $\gamma \leftarrow h \cdot \mathsf{BRW}_h(P_1||P_2||\ldots||P_m||T)$
3.  $\tau \leftarrow E_K(\gamma \oplus \alpha)$ ;
4.  for $j = 1$ to $m$
5.      $R_j \leftarrow E_K(\tau \oplus x^j\beta)$
6.      if $\mathsf{ty} = \mathsf{L}$
7.          $C_j \leftarrow R_j \oplus (1 \oplus x)P_j$
8.      else $C_j \leftarrow R_j \oplus xP_j$
9.      endif
10. endfor
11. return $(C_1||C_2||\ldots||C_m||\tau)$
```

```
Algorithm DCM-BRW.$\mathcal{D}_{h,K}^{T,\mathsf{ty}}(C_1, \ldots, C_m, \tau)$

1.  $\alpha = E_K(0); \beta \leftarrow E_K(1);$
2.  for $j = 1$ to $m$,
3.      $R_j \leftarrow E_K(\tau \oplus x^j\beta);$
4.      if $\mathsf{ty} = \mathsf{L}$
5.          $P_i \leftarrow (C_j \oplus R_j)(1 \oplus x)^{-1}$
6.      else $P_j \leftarrow (C_j \oplus R_j)x^{-1}$
7.      endif
8.  endfor
9.  $\gamma \leftarrow h \cdot \mathsf{BRW}_h(P_1||P_2||\ldots||P_m||T);$
10. $\tau' \leftarrow E_K(\gamma \oplus \alpha)$
11. if $\tau' = \tau$ return $(P_1, \ldots, P_m)$ else return $\perp$;
```

From property 2 of the BRW polynomials one can infer that to encrypt m block of messages DCM-BRW requires $\lfloor \frac{m+1}{2} \rfloor + 1$ multiplications and $\lg(m+1)$ squarings. Computing squares in binary fields are much more efficient than multiplication. In addition it requires $(m+3)$ block-cipher calls. Out of these $(m+3)$ block-cipher calls two can be pre-computed, but this would amount to the requirement of storage of key related materials which is not recommended. The construction requires two keys, h the key for the BRW polynomial and K the block-cipher key. Requirement of a single block-cipher key is what is important, as for multiple block-cipher keys one needs to have different key schedules which may make an implementation inefficient when implemented in hardware. One can probably generate the key h using the block-cipher key and still obtain a secure construction, but this would generally mean one more block-cipher call or a storage of key related material which is same as storage of an extra key. Also having a different key for the polynomial provides more flexibility during changing of keys. So, we decided to keep the block-cipher key independent of the key for the BRW polynomial.

The constructions requires two passes over the data, one for computing the tag τ and other for generating the ciphertext. The ciphertext is generated using a counter type mode of operation which can be parallelized, also α and β can be computed in parallel with the BRW polynomial. Thus the construction offers flexibility for efficient pipelined implementation. Also, for an efficient hardware implementation the only non-trivial blocks that are needed to be implemented are a finite field multiplier and a encryption only block-cipher. So, it is expected that a hardware implementation will have a small footprint.

10.3 Comparisons

To our knowledge we do not know of any other existing cryptographic scheme which provides the same functionality that is provided by a DCM. As mentioned before one can use a tweakable enciphering scheme to do secure backup, by writing the cipher in two different locations. Also, a certain class of DAE schemes can be used for constructing a DCM as mentioned in section 9. Hence, we compare the efficiency of DAE-BRW with existing tweakable enciphering schemes (TES) and DAE constructions. As DCM is completely a different primitive compared to either a TES or a DAE so these comparisons should be interpreted carefully, in particular we want to make explicit the following points before we present the real comparisons:

1. Any DCM scheme is not meant to be a competitor for any existing TES. The security guarantees that a TES provides are completely different from that of DCM. A TES is tag-less, but one needs to store a tag for any DCM scheme which amounts to extra storage. Moreover, the property of key-less recovery using the function g cannot be provided by a TES. Thus the comparison only point out the case of efficiency, but the difference in functionality and security guarantees should be considered while interpreting the efficiency comparisons.
2. Similarly, a DAE scheme which was proposed as a solution for the key-wrap problem [12] does not provide the security or functionality of a DCM. But as mentioned in section 9, a certain class of DAE schemes can be modified to obtain a DCM. Also the DCM-BRW can be modified to make it a DAE, the only modification required would be to output $R_j \oplus P_i$ (refer to the encryption algorithm in Fig. 3) instead of mixing R_j in two different ways with the plaintext to obtain two ciphertext. With this modification DCM-BRW would be converted into a secure DAE scheme which works on single block headers. But this is not the point that we are trying to focus here. The comparisons with DAE are meant to show that how efficient DCM-BRW would be compared to other possible constructions of DCM which can be easily derived by modifying existing DAE constructions. The efficiency in our construction compared to other DAE constructions comes from the use of BRW polynomials which require less multiplications.

Comparison with Tweakable Enciphering Schemes: Tweakable enciphering schemes which has been proposed as a solution to the low level disk encryption problem can be classified into three basic categories according to the way they are constructed. They are hash-ECB-hash, hash-counter-hash and encrypt-mask-encrypt types. For the first two types (for example TET, HCTR, HCH, XCB etc.) one uses two layers of polynomial hashes with an ECB or a counter mode and the last type (example CMC, EME) uses two encryption layers. To encrypt/decrypt a message/cipher of m blocks the first two types roughly require m block-cipher calls along with $2m$ multiplications. In [14] some new constructions of hash ECB hash type and hash counter hash type were proposed which uses a BRW polynomial to compute the hash, for these constructions about m multiplications are required. The encrypt-mask-encrypt type algorithms like EME [7] and CMC [6] require about $2m$ block cipher calls and no multiplication. In Table 1 we compare the number of operations required for tweakable enciphering schemes for fixed length messages which uses n bit tweaks with the number of operations required for DCM-BRW.

Table 1. Comparison of DCM-BRW with tweakable enciphering schemes for fixed length messages which uses n bit tweak. [BC]: Number of block-cipher calls; [M]: Number of multiplications, [BCK]: Number of blockcipher keys, [OK]: Other keys, including hash keys.

Mode	[BC]	[M]	[BCK]	[OK]
CMC [6]	$2m+1$	–	1	–
EME [7]	$2m+1$	–	1	–
XCB [10]	$m+1$	$2(m+3)$	3	2
HCTR [16]	m	$(2m+1)$	1	1
HCHfp [3]	$m+2$	$2(m-1)$	1	1
TET [5]	$m+1$	$2m$	2	3
Constructions in [14] using normal polynomials	$m+1$	$2(m-1)$	1	1
Constructions in [14] using BRW polynomials	$m+1$	$2+2\lfloor(m-1)/2\rfloor$	1	
DCM-BRW	$m+3$	$1+\lfloor(m+1)/2\rfloor$	1	1

From Table 1 we can see that encrypting with DCM-BRW would be much more efficient than encrypting by any of the existing tweakable enciphering schemes. But, it is to be noted that the security guarantee that a tweakable enciphering scheme provides is very different from that provided by a DCM, hence this comparison needs to be appropriately interpreted.
Comparison with Deterministic Authenticated Encryption Schemes: As mentioned in Section 9, deterministic authenticated encryption (DAE) schemes which uses counter mode of operation can be easily converted into a DCM scheme by only using additional xor operations. There are three DAE schemes reported till date. The SIV mode [12] uses CMAC along with the counter mode, and [8, 9] uses a variant of a polynomial hash with a counter mode. All these schemes can be reworked to construct a DCM. But as evident from Table 2 DCM-BRW would be much more efficient than any of them. For the DAE schemes the operation counts shown in Table 2 are based on only one block of tweak.

Table 2. Comparison between DCM-BRW and DAE schemes for encrypting m blocks of messages. In the DAE schemes the operation counts are based on only one block of tweak. [BC]: Number of block-cipher calls; [M]: Number of multiplications, [BCK]: Number of blockcipher keys, [OK]: Other keys, including hash keys.

Mode	$[BC]$	$[M]$	[BCK]	[OK]
SIV [12]	$2m+3$	–	2	–
HBS [9]	$m+2$	$m+3$	1	–
BTM [8]	$m+3$	m	1	–
DCM-BRW	$m+3$	$1+\lfloor(m+1)/2\rfloor$	1	1

11 Security of DCM-BRW

The following theorem specifies the security of DCM-BRW.

Theorem 3. *Let $\Pi = Perm(n)$. Let $\mathcal{A}$ be an adversary attacking* DCM-BRW$[\Pi]$ *who asks q queries, then*

$$\mathbf{Adv}^{\text{dcm-priv}}_{\text{DCM-BRW}[\Pi]}(\mathcal{A}) \leq \frac{14m^2q^2}{2^n} \tag{6}$$

$$\mathbf{Adv}^{\text{dcm-auth}}_{\text{DCM-BRW}[\Pi]}(\mathcal{A}) \leq \frac{1}{2^n} + \frac{18m^2q^2}{2^n} \tag{7}$$

Theorem 4. *Let $E : \mathcal{K} \times \{0,1\}^n \rightarrow \{0,1\}^n$ be a block-cipher secure in the PRP sense. Let $\mathcal{A}$ be an adversary attacking* DCM-BRW$[E]$ *who asks q queries, then their exists an adversary $\mathcal{A}'$ such that*

$$\mathbf{Adv}^{\text{dcm-priv}}_{\text{DCM-BRW}[\Pi]}(\mathcal{A}) \leq \frac{14m^2q^2}{2^n} + \mathbf{Adv}^{\text{prp}}_{E}(\mathcal{A}') \tag{8}$$

$$\mathbf{Adv}^{\text{dcm-auth}}_{\text{DCM-BRW}[E]}(\mathcal{A}) \leq 2\mathbf{Adv}^{\text{prp}}_{E}(\mathcal{A}') + \frac{1}{2^n} + \frac{18m^2q^2}{2^n} \tag{9}$$

where $\mathcal{A}'$ asks $O(q)$ queries and run for time $t + O(q)$ where t is the running time of $\mathcal{A}$.

Transition from Theorem 3 to Theorem 4 is quite standard, we provide the proof of Theorem 3 in Appendix B. The proof uses the standard game playing technique as used in [3, 6, 7], particularly it resembles the proof presented in [9].

12 Another Possible Use of a DCM: Secret Sharing with Authority

Here we describe a different scenario other than secure backup where DCM can be useful. Consider that the military general (A) wants to share a secret with his subordinate (B) who is to join the battalion in the field. A should be able to recover the secret without the consent of B but B should not be able to recover the secret without the consent of A. One of the solutions to this may be that A encrypts the secret and gives it to B and reveals the key to B when A feels that B needs to know the secret. But this involves revealing a key, which can have important consequences when A and B are geographically separated. We propose a solution through a secure DCM as follows. For a secret M, A computes the encryption of M using both the types $\mathsf{ty} = \mathsf{R}$ and $\mathsf{ty} = \mathsf{L}$ and gives one of the cipher without the tag to B. A can decrypt the secret using his key and tag at any time (s)he desires and when (s)he wants B to know the secret then (s)he reveals his/her cipher to B and thus B can recover the secret using the recovery function g.

13 Conclusion

We studied a new type of cryptographic scheme called the Double Ciphertext Mode. We gave a generic construction and a particular construction using BRW polynomials. We also explored some applications where this mode can be useful. We identify some interesting problems which we plan to investigate in the context of DCM in future:

1. Our construction does not work for messages whose lengths are not multiples of the block length of the block-cipher, a construction which overcomes this restriction may be interesting.
2. A generalization of the DCM scheme into a multiple ciphertext mode where more than two ciphertexts are produced may be interesting.

References

1. Daniel J. Bernstein. Polynomial evaluation and message authentication, 2007. http://cr.yp.to/papers.html#pema.
2. Debrup Chakraborty and Palash Sarkar. A New Mode of Encryption Providing a Tweakable Strong Pseudo-random Permutation. In Matthew J. B. Robshaw, editor, *Fast Software Encryption 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 293–309. Springer, 2006.
3. 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.

4. Niels Ferguson. AES-CBC + Elephant diffuser: A Disk Encryption Algorithm for Windows Vista. Microsoft white paper, 2006. http://download.microsoft.com/download/0/2/3/0238acaf-d3bf-4a6d-b3d6-0a0be4bbb36e/BitLockerCipher200608.pdf.
5. Shai Halevi. Invertible universal hashing and the TET encryption mode. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *Lecture Notes in Computer Science*, pages 412–429. Springer, 2007.
6. Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 482–499. Springer, 2003.
7. Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer, 2004.
8. Tetsu Iwata and Kan Yasuda. BTM: A single-key, inverse-cipher-free mode for deterministic authenticated encryption. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *Selected Areas in Cryptography*, volume 5867 of *Lecture Notes in Computer Science*, pages 313–330. Springer, 2009.
9. Tetsu Iwata and Kan Yasuda. HBS: A single-key mode of operation for deterministic authenticated encryption. In Orr Dunkelman, editor, *FSE*, volume 5665 of *Lecture Notes in Computer Science*, pages 394–415. Springer, 2009.
10. David A. McGrew and Scott R. Fluhrer. The extended codebook (XCB) mode of operation. Cryptology ePrint Archive, Report 2004/278, 2004. http://eprint.iacr.org/.
11. Michael O. Rabin and Shmuel Winograd. Fast evaluation of polynomials by rational preparation. *Communications on Pure and Applied Mathematics*, 25:433458, 1972.
12. Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 373–390. Springer, 2006.
13. Palash Sarkar. Improving upon the TET mode of operation. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *ICISC*, volume 4817 of *Lecture Notes in Computer Science*, pages 180–192. Springer, 2007.
14. Palash Sarkar. Efficient tweakable enciphering schemes from (block-wise) universal hash functions. *IEEE Transactions on Information Theory*, 55(10):4749–4760, 2009.
15. Palash Sarkar. Pseudo-random functions and parallelizable modes of operations of a block cipher. Cryptology ePrint Archive, Report 2009/217, 2009. http://eprint.iacr.org/.
16. Peng Wang, Dengguo Feng, and Wenling Wu. HCTR: A Variable-Input-Length Enciphering Mode. In Dengguo Feng, Dongdai Lin, and Moti Yung, editors, *CISC*, volume 3822 of *Lecture Notes in Computer Science*, pages 175–188. Springer, 2005.

A Proof of Theorem 1

To prove the security of DCM$[\Upsilon, \Pi]$ we replace the pseudorandom function F in Fig. 1 by a function ρ chosen uniformly at random from $\Upsilon = \text{Func}(mn+n, n)$, and the block cipher E by a permutation π chosen uniformly at random from $\Pi = \text{Perm}(n)$. We call the encryption function of DCM$[\Upsilon, \Pi]$ as $\mathcal{E}_{\rho,\pi}$. We consider that $\mathcal{A}$ interacts with the game DCM1 as depicted in Fig. 4. Notice that in DCM1, the proper oracles for $\mathcal{A}$ are provided, the random function ρ and the random permutation π are constructed on the fly using the subroutines Ch-$\pi(.)$ and Ch-$\rho(.)$ respectively. The domain and range sets of the permutation π are maintained in the sets $Domain_\pi$ and $Range_\pi$ respectively, and appropriate checks are made so that the subroutine Ch-$\pi(X)$ indeed behaves like a permutation. No checks are necessary in case of Ch-$\rho(X)$, as according to the query restrictions of $\mathcal{A}$, Ch-$\rho(.)$ is always called with a fresh (new) input.

For a game G, let $\Pr[\mathcal{A}^G \Rightarrow 1]$ denote the probability that $\mathcal{A}$ outputs 1 by interacting with the game G. Then we have,

$$\Pr\left[\mathcal{A}^{\mathcal{E}_{\rho,\pi}(.,.,.)} \Rightarrow 1\right] = \Pr[\mathcal{A}^{\text{DCM1}} \Rightarrow 1]. \tag{10}$$

We modify game DCM1 by deleting the boxes entries in Fig. 4 and call the modified game as DCM2. By deleting the boxed entries it cannot be guaranteed that Ch-π is a permutation as though we do the consistency checks but we do not reset the values of Y in Ch-$\pi(.)$. Then the games DCM1 and DCM2 are identical except when the bad flag is set, thus we have

$$\Pr[\mathcal{A}^{\text{DCM1}} \Rightarrow 1] - \Pr[\mathcal{A}^{\text{DCM2}} \Rightarrow 1] \leq \Pr[\mathcal{A}^{\text{DCM2}} \text{ sets } \mathsf{bad}] \tag{11}$$

Fig. 4. Games DCM1 and DCM2: DCM2 is the game without the boxed entries in the subroutine Ch-$\pi()$

```
     Subroutine Ch-$\pi(X)$
11.     $Y \xleftarrow{\$} \{0,1\}^n$; **if** $Y \in Range_\pi$ **then** bad $\leftarrow$ true; $\boxed{Y \xleftarrow{\$} \overline{Range_\pi}}$; **endif**;
12.     **if** $X \in Domain_\pi$ **then** bad $\leftarrow$ true; $\boxed{Y \leftarrow \pi(X)}$; **endif**
13.     $\pi(X) \leftarrow Y$; $Domain_\pi \leftarrow Domain_\pi \cup \{X\}$; $Range_\pi \leftarrow Range_\pi \cup \{Y\}$; **return**$(Y)$;

     Subroutine Ch-$\rho(M)$
14.     $Y \xleftarrow{\$} \{0,1\}^n$;
15.     **return**$(Y)$;

     Initialization:
16.     $Domain_\pi \leftarrow Range_\pi \leftarrow \emptyset$;
17.     **for** all $X \in \{0,1\}^n$, $\pi(X) =$ **undefined endfor**
18.     bad $\leftarrow$ false
```

```
Respond to the $s^{th}$ encryption query of $\mathcal{A}$, $(T^s; P_1^s, P_2^s, \ldots, P_m^s; \mathsf{ty}^s)$ as follows:
101. $\tau^s \leftarrow \rho(P_1^s||P_2^s||\ldots||P_m^s||T^s)$;
102. **for** $i = 1$ to $m$,
103.    $R_i^s \leftarrow \pi(\tau^s \oplus \mathsf{bin}_n(i))$;
104.    **if** $\mathsf{ty}^s = \mathsf{L}$ **then**
105.       $C_i^s \leftarrow R_i^s \oplus (1 \oplus x)P_i^s$;
106.    **else**
107.       $C_i^s \leftarrow R_i^s \oplus xP_i^s$;
108.    **endif**
109. **endfor**
110. **return** $(C_1^s||C_2^s||\ldots||C_{m^s}^s||\tau^s)$
```

Fig. 5. Game DCM3: D_π and R_π are multisets, which are initially empty. We assume that $\mathcal{A}$ makes q queries.

```
Respond to the $s^{th}$ encryption query $(T^s; P_1^s, P_2^s, \ldots, P_{m^s}^s; \mathsf{ty}^s)$ as follows
        $C_1^s||C_2^s||\ldots||C_m^s||\tau^s \xleftarrow{\$} \{0,1\}^{(m+1)n}$
        **Return** $C_1^s||C_2^s||\ldots||C_{m^s}^s||\tau^s$
**Finalization:**
FIRST PHASE
**for** $s = 1$ to $q$
    **for** $i = 1$ to $m$,
        $D_\pi \leftarrow D_\pi \cup \{\tau^s \oplus \mathsf{bin}_n(i)\}$
        **if** $\mathsf{ty}^s = \mathsf{L}$, **then** $\delta_i^s \leftarrow (1 \oplus x)P_i^s$;
        **else** $\delta_i^s \leftarrow xP_i^s$
        **end if**
        $R_\pi \leftarrow R_\pi \cup \{C_i^s \oplus \delta_i^s\}$
    **end for**
**end for**
SECOND PHASE
    bad $\leftarrow$ false;
    **if** (some value occurs more than once in $D_\pi$) **then** bad $=$ true **end if**;
    **if** (some value occurs more than once in $R_\pi$) **then** bad $=$ true **end if**.
```

In the game DCM2, $\mathcal{A}$ always gets random strings as a response to an encryption query. This can be seen in lines 101 and 103 of the game DCM3, where τ^s and R_i^s $(1 \leq i \leq m)$ gets set to a random n bit strings. Thus,

$$\Pr[\mathcal{A}^{\mathrm{DCM2}} \Rightarrow 1] = \Pr\left[\mathcal{A}^{\$(.,.,.)} \Rightarrow 1\right]. \tag{12}$$

So using Equations (10), (11) and (12) we have

$$\begin{aligned}\mathbf{Adv}^{\text{dcm-priv}}_{\mathrm{DCMG}[\Upsilon,\Pi]}(\mathcal{A}) &= \Pr\left[\mathcal{A}^{\mathcal{E}_{\rho,\pi}(.,.,.)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(.,.,.)} \Rightarrow 1\right] \\ &\leq \Pr[A^{\mathrm{DCM2}} \text{ sets } \mathsf{bad}] \end{aligned} \tag{13}$$

Now, we make a game which no more use subroutines Ch-π and Ch-ρ, and immediately returns random strings to the adversary in response to his encryptions queries. Later we maintain the multi-sets D_π and R_π where we list the elements that were supposed to be inputs and outputs of the permutation. Finally we check the collisions in the multi-sets D_π and R_π, and if there is any collision we set the bad flag to true. We call this game as DCM3, which is shown in Fig. 5. The games DCM2 and DCM3 are indistinguishable to the adversary, because in both cases it gets random strings in response to his queries. Hence the probability with which DCM2 sets bad is same as the probability with which DCM3 sets bad. Thus we get,

$$\Pr[\mathcal{A}^{\mathrm{DCM2}} \text{ sets } \mathsf{bad}] = \Pr[\mathcal{A}^{\mathrm{DCM3}} \text{ sets } \mathsf{bad}] \tag{14}$$

Thus from equations (13) and (14) we obtain

$$\mathbf{Adv}^{\text{dcm-priv}}_{\mathrm{DCMG}[\Upsilon,\Pi]}(\mathcal{A}) \leq \Pr[\mathcal{A}^{\mathrm{DCM3}} \text{ sets } \mathsf{bad}] \tag{15}$$

Now our goal would be to bound $\Pr[\mathcal{A}^{\mathrm{DCM3}}$ sets $\mathsf{bad}]$. We can see in Game DCM3 that the bad flag is set when there is a collision in either D_π or R_π. So if COLD_π and COLR_π denote the events of a collision in D_π and R_π respectively then we have

$$\Pr[\mathcal{A}^{\mathrm{DCM3}} \text{ sets } \mathsf{bad}] = \Pr[\mathsf{COLD}_\pi] + \Pr[\mathsf{COLR}_\pi]. \tag{16}$$

Now, we shall bound the collision probabilities in the domain and range sets. From Fig. 5 we see that $D_\pi = \{\tau^s \oplus \mathsf{bin}_n(i) : 1 \leq i \leq m, 1 \leq s \leq q\}$, where τ^s is an element chosen uniformly at random from $\{0,1\}^n$. And, $R_\pi = \{C_i^s \oplus \delta_i^s : 1 \leq i \leq m, 1 \leq s \leq q\}$, where $\delta_i^s = (1 \oplus x)P_i$ when $\mathsf{ty}^s = \mathsf{L}$ and $\delta_i^s = xP_i$ when $\mathsf{ty}^s = \mathsf{R}$, and C_i^s is chosen uniformly at random from $\{0,1\}^n$.

Thus, the probability of collisions between two elements in D_π is given by $\Pr[\mathsf{COLD}_\pi] = \Pr[\tau^s \oplus \mathsf{bin}_n(i) = \tau^t \oplus \mathsf{bin}_n(j)]$ where s and t are two different queries, $1 \leq i, j \leq m$, this probability is at most 2^{-n}, as τ^s is always a string chosen uniformly at random from $\{0,1\}^n$. The number of elements in D_π is equal to qm, thus we have

$$\Pr[\mathsf{COLD}] \leq \binom{mq}{2}\frac{1}{2^n} < \frac{m^2q^2}{2^{n+1}} \tag{17}$$

As each C_i^s is chosen uniformly at random from $\{0,1\}^n$ and R_π contains mq elements, hence we have

$$\Pr[\mathsf{COLR}] \leq \frac{m^2q^2}{2^{n+1}} \tag{18}$$

Finally we have

$$\mathbf{Adv}^{\text{dcm-priv}}_{\mathrm{DCMG}[\Upsilon,\Pi]}(\mathcal{A}) \leq \Pr[\mathsf{COLD}] + \Pr[\mathsf{COLR}] \leq \frac{m^2q^2}{2^n} \tag{19}$$

which completes the proof for the privacy bound.

Proving the authentication bound: To bound the authentication advantage, we give an oracle access of the random permutation π to the adversary $\mathcal{A}$. We denote the adversary $\mathcal{A}$ with an oracle access to π as $\mathcal{A}(\pi)$. Thus we can say that

$$\mathbf{Adv}^{\text{dcm-auth}}_{\text{DCMG}[\Upsilon,\Pi]}(\mathcal{A}) \leq \mathbf{Adv}^{\text{dcm-auth}}_{\text{DCMG}[\Upsilon,\Pi]}(\mathcal{A}(\pi)),$$

as $\mathcal{A}$ without oracle access to π can do no better than $\mathcal{A}(\pi)$. $\mathbf{Adv}^{\text{dcm-auth}}_{\text{DCMG}[\Upsilon,\Pi]}(\mathcal{A}(\pi))$ is bounded by the probability that $\mathcal{A}(\pi)$ produces a forgery $(\mathcal{C}, T, \mathsf{ty})$, such that $\mathcal{D}_{\rho,\pi}(\mathcal{C}, T, \mathsf{ty}) \neq \perp$. A forgery $(C_1||\ldots||C_m||\tau, T, \mathsf{ty})$ can be successful if $\rho(X||T) = \tau$, where $X = X_1||X_2||\ldots||X_m$, and $X_i = (1 \oplus x)^{-1}[C_i \oplus \pi(\tau \oplus \mathsf{bin}_n(i))]$ when $\mathsf{ty} = \mathsf{L}$ and $X_i = x^{-1}[C_i \oplus \pi(\tau \oplus \mathsf{bin}_n(i))]$ when $\mathsf{ty} = \mathsf{R}$.

As $\mathcal{A}(\pi)$ has an oracle access to π, so it can compute X for any chosen $\mathcal{C}$. Thus, when it produces a forgery $(T, \mathcal{C}, \mathsf{ty})$ it knows what would be the input to the function ρ and also the target output for his forgery. But according to the query restrictions, the adversary never produces $(T, C||\tau, \mathsf{ty})$ as a forgery if it obtained $C||\tau$ as an reply to an encryption query of the form (T, X, ty). Also it does not produce $(T, C||\tau, \mathsf{ty})$ as a forgery if it obtained $\tilde{C}||\tau$ as a response to its query $(T, X, \bar{\mathsf{ty}})$, where $g(C, \tilde{C}) = X$. Thus either the input $X||T$ to ρ is new or the output τ is new, thus the probability that $\rho(X||T) = \tau$ is at most $1/2^n$, hence we have

$$\mathbf{Adv}^{\text{dcm-auth}}_{\text{DCMG}[\Upsilon,\Pi]}(\mathcal{A}) \leq \mathbf{Adv}^{\text{dcm-auth}}_{\text{DCMG}[\Upsilon,\Pi]}(\mathcal{A}(\pi)) \leq \frac{1}{2^n}. \tag{20}$$

Which completes the proof. ∎

B Proof of Theorem 3

Before proving Theorem 3 we shall establish two results which would be useful for us in proving the theorem.

Lemma 1. *Let $X = (X_1, X_2, \ldots, X_m), X' = (X'_1, X'_2, \ldots, X'_m) \in [GF(2^n)]^m$ such that $X \neq X'$. Let $Y = h\mathsf{BRW}_h(X_1, X_2, \ldots, X_m)$ and $Y' = h\mathsf{BRW}_h(X'_1, X'_2, \ldots, X'_m)$. Then for every fixed $\delta \in GF(2^n)$*

$$\Pr[Y \oplus Y' = \delta] \leq \frac{2m}{2^n},$$

The probability is taken over the random choice of h.

This easily follows from property 3 of the BRW polynomials as stated in Section 10.1 [15].

For a permutation π The following result suggests that $f_{h,\pi}(.,.)$ is a pseudorandom function.

Lemma 2. *Let $\pi \xleftarrow{\$} \text{Perm}(n)$, and $h \xleftarrow{\$} \{0,1\}^n$. Let $f_{h,\pi}(.,.)$ be a function such that given an input $(X, T) \in \{0,1\}^{nm} \times \{0,1\}^n$, it returns $\pi(h\mathsf{BRW}_h(X||T) \oplus \pi(0))$. Let $\mathcal{B}$ be an prf adversary attacking $f_{h,\pi}(.,.)$ and $\mathcal{B}$ asks a total of q queries. Then*

$$\begin{aligned}\mathbf{Adv}^{\text{prf}}_{f}(\mathcal{B}) &= \Pr\left[\mathcal{B}^{f_{h,\pi}(.,.)} \Rightarrow 1\right] - \Pr\left[\rho \xleftarrow{\$} \text{Func}(mn+n, n) : \mathcal{A}^{\rho(.)} \Rightarrow 1\right] \\ &< \frac{4mq^2}{2^n}.\end{aligned}$$

Proof. We use the game playing technique. Consider the games F0 and F1 as shown in Fig. 6. The difference between the two games is that in F0 outputs of the permutation π is used which is constructed in the fly by the subroutine Ch-π() as shown Fig. 4. In game F1, the boxed entries of Ch-π() as shown Fig. 4 are removed. These games run in the same way until the bad flag is set. Hence,

$$\Pr\left[\mathcal{B}^{\text{F0}} \Rightarrow 1\right] - \Pr\left[\mathcal{B}^{\text{F1}} \Rightarrow 1\right] \leq \Pr[\mathcal{B}^{\text{F1}} \text{ sets } \mathsf{bad}].$$

Fig. 6. Games F0 and F1. In game F0 the subroutine Ch-$\pi()$ as in game DCM1 (i.e. with the boxed entries in Fig. 4) is used. In game F1 the subroutine Ch-$\pi()$ as in game DCM2 is used (i.e. without the boxed entries in Fig. 4).

```
Initialization:
01. **for** all $X \in \{0,1\}^n$ $\pi(X) =$ **undefined endfor**
02. $EZ \xleftarrow{\$} \{0,1\}^n$; $Domain_\pi \leftarrow Domain_\pi \cup \{0\}$; $Range_\pi \leftarrow Range_\pi \cup \{EZ\}$
03. $\mathsf{bad} \leftarrow \mathsf{false}$

Respond to the $s^{th}$ query $(X_1^s||X_2^s||\dots||X_m^s, T^s)$ of $\mathcal{B}$ as follows:
11.   $t^s \leftarrow \mathsf{BRW}_h(X_1^s||X_2^s||\dots||X_m^s||T^s)$;
12.   $\tau^s \leftarrow$ Ch-$\pi(t^s \oplus EZ)$
13.   **Return** $(\tau^s)$
```

Also, the responses obtained by $\mathcal{B}$ in game F1 are elements selected uniformly at random from $\{0,1\}^n$. As $\mathcal{B}$ does not repeat a query, hence the outputs received by $\mathcal{B}$ in game F1 would be indistinguishable from those obtained from a function sampled uniformly at random from Func$(mn+n,n)$. Thus,

$$\Pr\left[\mathcal{B}^{\mathrm{F1}} \Rightarrow 1\right] = \Pr\left[\rho \xleftarrow{\$} \mathrm{Func}(mn+n,n) : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1\right].$$

Hence we have

$$\mathbf{Adv}^{\mathrm{prf}}_{f\text{-BRW}}(\mathcal{B}) \leq \Pr[\mathcal{B}^{\mathrm{F1}} \text{ sets } \mathsf{bad}].$$

Following the construction of Ch-$\pi()$, the bad flag in game F1 is set when there is a collision in the set $Domain_\pi$ or the set $Range_\pi$. Let us list all the values that gets in the domain and range sets in the multisets $\mathcal{SS}$ and $\mathcal{RR}$. Then we would have, $\mathcal{SS} = \{0\} \cup \{t^s \oplus EZ : 1 \leq s \leq q\}$ and $\mathcal{RR} = \{EZ\} \cup \{\tau^s : 1 \leq s \leq q\}$. Let COL be the event that there is a collision in $\mathcal{SS}$ or in $\mathcal{RR}$. Then,

$$\Pr[\mathcal{B}^{\mathrm{F1}} \text{ sets } \mathsf{bad}] = \Pr[\mathsf{COL}].$$

Note that for two distinct queries u, v, we have $t^u = h\mathsf{BRW}_h(X_1^u||X_2^u||\dots||X_m^u||T^u)$, and $t^v = h\mathsf{BRW}_h(X_1^v||X_2^v||\dots||X_m^v||T^v)$ where $X_1^u||X_2^u||\dots||X_m^u||T^u \neq X_1^v||X_2^v||\dots||X_m^v||T^v$. Thus from Lemma 1 we have $\Pr[t^u = t^v] = 2(m+1)/2^n$. Also $\Pr[t^u \oplus EZ = 0] = 1/2^n$ as EZ is a random element of $\{0,1\}^n$, and $\Pr[\tau^u = \tau^v] = \Pr[\tau^u = EZ] = 1/2^n$, as all τ^u, τ^v and EZ are selected uniformly at random from $\{0,1\}^n$. Hence combining the above facts we get

$$\begin{aligned}\Pr[\mathsf{COL}] &= \binom{q}{2}\frac{2(m+1)}{2^n} + \frac{q}{2^n} + \binom{q+1}{2}\frac{1}{2^n} \\ &< \frac{4mq^2}{2^n}.\end{aligned}$$

Thus, we have

$$\mathbf{Adv}^{\mathrm{prf}}_{f}(\mathcal{B}) \leq \frac{4mq^2}{2^n}, \tag{21}$$

which completes the proof. ■

Proof of Theorem 3: To prove the security of DCM-BRW$[\Pi]$ we replace the the block cipher E in the construction of Fig. 3 by a permutation π chosen uniformly at random from $\Pi = \mathrm{Perm}(n)$. We call the encryption function of DCM-BRW$[\Pi]$ as $\mathcal{E}_{h,\pi}$, where $h \xleftarrow{\$} \{0,1\}^n$ is the key for the BRW polynomial. We prove the privacy bound first. We consider the same game playing technique as used in the proof of Theorem 1. We briefly discuss the games below:

1. Game G0: In G0 (shown in Fig. 7) the block-cipher is replaced by the random permutation π. The permutation π is constructed on the fly keeping record of the domain and range sets as done

in the sub-routine Ch-π in Fig. 7. Thus, G0 provide the proper encryption oracle to $\mathcal{A}$. Thus we have:

$$\Pr\left[\mathcal{A}^{\mathcal{E}_{h,\pi}(.,.,.)} \Rightarrow 1\right] = \Pr[\mathcal{A}^{\text{G0}} \Rightarrow 1]. \tag{22}$$

2. Game G1: Fig. 7 with the boxed entries removed represents the game G1. In G1 it is not guaranteed that Ch-π behaves like a permutation, but the games G0 and G1 are identical until the bad flag is set. Thus we have

$$\Pr[\mathcal{A}^{\text{G0}} \Rightarrow 1] - \Pr[\mathcal{A}^{\text{G1}} \Rightarrow 1] \leq \Pr[\mathcal{A}^{\text{G1}} \text{ sets } \mathsf{bad}] \tag{23}$$

Note that in G1 the adversary gets random strings as output irrespective of his queries. Hence,

$$\Pr[\mathcal{A}^{\text{G1}} \Rightarrow 1] = \Pr[\mathcal{A}^{\$(.,.,.)} \Rightarrow 1] \tag{24}$$

3. Game G2: In Game G2 (shown in Fig. 8) we do not use the subroutine Ch-π any more but return random strings immediately after the $\mathcal{A}$ asks a query. Later we keep track of the elements that would have got in the domain and range sets of the permutation π in multi-sets $\mathcal{S}$ and $\mathcal{R}$. We set the bad flag when there is a collision in either $\mathcal{S}$ or $\mathcal{R}$. For the adversary the games G1 and G2 are identical. So,

$$\Pr[\mathcal{A}^{\text{G1}} \Rightarrow 1] = \Pr[\mathcal{A}^{\text{G2}} \Rightarrow 1] \tag{25}$$

and

$$\Pr[\mathcal{A}^{\text{G1}} \text{ sets } \mathsf{bad}] = \Pr[\mathcal{A}^{\text{G2}} \text{ sets } \mathsf{bad}] \tag{26}$$

Fig. 7. Games G0 and G1

```
    Subroutine Ch-$\pi(X)$
01.   $Y \xleftarrow{\$} \{0,1\}^n$; if $Y \in Range_\pi$ then bad $\leftarrow$ true; $\boxed{Y \xleftarrow{\$} \overline{Range_\pi}}$; endif;
02.   if $X \in Domain_\pi$ then bad $\leftarrow$ true; $\boxed{Y \leftarrow \pi(X)}$; endif
03.   $\pi(X) \leftarrow Y$; $Domain_\pi \leftarrow Domain_\pi \cup \{X\}$; $Range_\pi \leftarrow Range_\pi \cup \{Y\}$; return$(Y)$;

    Initialization:
11.   for all $X \in \{0,1\}^n$ $\pi(X)$ = undefined endfor
12.   $EZ \xleftarrow{\$} \{0,1\}^n$; $\pi(0) = EZ$;
13.   $Domain_\pi \leftarrow \{0\}$; $Range_\pi \leftarrow \{EZ\}$;
14.   $EO \xleftarrow{\$} \{0,1\}^n \setminus \{EZ\}$; $\pi(1) = EO$;
15.   $Domain_\pi \leftarrow Domain_\pi \cup \{1\}$; $Range_\pi \leftarrow Range_\pi \cup \{EO\}$;
16.   bad = false

Respond to the $s^{th}$ encryption query $(T^s; P_1^s, P_2^s, \ldots, P_m^s; ty^s)$ as follows:
101. $\gamma^s \leftarrow h \cdot BRW(P_1^s||P_2^s||\ldots||P_m^s||T^s)$;
102. $\tau^s \leftarrow$ Ch-$\pi(\gamma^s \oplus EZ)$;
103. for $i = 1$ to $m$,
104.   $R_i^s \leftarrow$ Ch-$\pi(\tau^s \oplus x^i EO)$;
105.   if $ty^s = L$ then $C_i^s \leftarrow R_i^s \oplus xP_i^s$;
106.   else $C_i^s \leftarrow R_i^s \oplus (x \oplus 1)P_i^s$;
107.   endif
108. endfor
109. Return $(C_1^s||C_2^s||\ldots||C_{m^s}^s||\tau^s)$
```

Hence, using Eqs. (22), (23), (24), (25) and (26), we have

$$\Pr\left[\mathcal{A}^{\mathcal{E}_{h,\pi}(.,.,.)} \Rightarrow 1\right] - \Pr[\mathcal{A}^{\$(.,.,.)} \Rightarrow 1] \leq \Pr[\mathcal{A}^{\mathrm{G2}} \text{ sets } \mathsf{bad}].$$

According to the definition of the privacy advantage of $\mathcal{A}$, we have

$$\mathbf{Adv}^{\text{dcm-priv}}_{\text{DCM-BRW}[\mathit{\Pi}]}(\mathcal{A}) \leq \Pr[\mathcal{A}^{\mathrm{G2}} \text{ sets } \mathsf{bad}] \tag{27}$$

Now we need to bound $\Pr[\mathcal{A}^{\mathrm{G2}}$ sets $\mathsf{bad}]$. The elements in the multi-sets $\mathcal{S}$ and $\mathcal{R}$ would be

$$\mathcal{S} = \{0, 1\} \cup \{\gamma^s \oplus EZ : 1 \leq s \leq q\} \cup \{\tau^s \oplus x^i EO : 1 \leq i \leq m, 1 \leq s \leq q\} \tag{28}$$

$$\mathcal{R} = \{EZ, EO\} \cup \{\tau^s : 1 \leq s \leq q\} \cup \{C_i^s \oplus \delta_i : 1 \leq i \leq m, 1 \leq s \leq q\}, \tag{29}$$

where $\delta_i^s = xP_i$ when $\mathsf{ty}^s = \mathsf{L}$ and $\delta_i^s = (x \oplus 1)P_i$ when $\mathsf{ty}^s = \mathsf{R}$. Let COLLD be the event that there is a collision in $\mathcal{S}$ and COLLR be the event that there is a collision in $\mathcal{R}$ Using the facts that $\gamma^s = h\mathsf{BRW}_h(P_1^s||P_2^s||\ldots||P_m^s||T^s)$ and EZ, EO, τ^s, C_i^s are random elements of $\{0,1\}^n$, we have

$$\begin{aligned}\Pr[\mathsf{COLLD}] &\leq \frac{2q}{2^n} + \frac{2mq}{2^n} + \binom{q}{2}\frac{(2m+2)}{2^n} + \binom{mq}{2}\frac{1}{2^n} + \frac{2mq^2(m+1)}{2^n} \\ &= \frac{1}{2^n}\left(\frac{5}{2}m^2q^2 + 3mq^2 + \frac{mq}{2} + q + q^2\right)\end{aligned}$$

and

$$\begin{aligned}\Pr[\mathsf{COLLR}] &= \binom{(mq+q+2)}{2}\frac{1}{2^n} \\ &= \frac{(m^2q^2 + 2mq^2 + 3mq + q^2 + 3q + 2)}{2^{n+1}}\end{aligned}$$

Then we have

$$\begin{aligned}\Pr[\mathcal{A}^{\mathrm{G2}} \text{ sets } \mathsf{bad}] &= \Pr[\mathsf{COLLD}] + \Pr[\mathsf{COLLD}] \\ &< \frac{14m^2q^2}{2^n}\end{aligned} \tag{30}$$

This completes the proof of the privacy bound.

Proving the authentication bound: To prove the authenticity bound, let $\mathcal{A}$ be an adversary which tries to break authenticity of DCM-BRW[$\mathit{\Pi}$]. Let $\mathcal{B}$ be an adversary attacking authenticity of $f_{\pi,h}$ (see Lemma 2 for the description of $f_{\pi,h}$). $\mathcal{B}$ has an oracle access to $f_{h,\pi}(.,.)$, additionally let it have access to another oracle $\mathcal{P}(.,.)$ which on input (X, i) returns $\pi(X \oplus x^i\pi(1))$. With access to these two oracles $\mathcal{B}$ can run $\mathcal{A}$ by answering its queries in the usual manner. When $\mathcal{A}$ outputs a forgery $(T, Y_1||Y_2||\ldots||Y_m||\tau, \mathsf{ty})$, $\mathcal{B}$ computes X_i, $1 \leq i \leq m$, using the oracle $\mathcal{P}(.,.)$ as follows

$$X_i = \begin{cases} (\mathcal{P}(t,i) \oplus Y_i)(1 \oplus x)^{-1} & \text{if } \mathsf{ty} = \mathsf{L} \\ (\mathcal{P}(t,i) \oplus Y_i)x^{-1} & \text{if } \mathsf{ty} = \mathsf{R} \end{cases}$$

and outputs $(X_1||X_2||\ldots||X_m||T, \tau)$ as its forgery. Thus we have that

$$\begin{aligned}\mathbf{Adv}^{\text{dcm-auth}}_{\text{DCM-BRW}[\mathit{\Pi}]}(\mathcal{A}) &= \Pr[\mathcal{A}^{\mathcal{E}_{h,\pi}(.,.,.)} \text{ forges}] \\ &\leq \Pr[\mathcal{B}^{f_{h,\pi}(.),\mathcal{P}(.,.)} \text{ forges}]\end{aligned} \tag{31}$$

Now, we replace this oracle $\mathcal{P}(.,.)$ with an oracle $\$(.,.)$ which returns strings chosen uniformly at random from $\{0,1\}^n$ on a query (X, i). The difference of the real oracle $\mathcal{P}$ and the oracle $\$(.,.)$

Fig. 8. Game G2: $\mathcal{S}$ and $\mathcal{R}$ are multisets.

```
Initialization:
    $\mathcal{S} \leftarrow \mathcal{R} \leftarrow \emptyset$;
    $EZ \xleftarrow{\$} \{0,1\}^n$; $EO \xleftarrow{\$} \{0,1\}^n \setminus \{EZ\}$;
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{0,1\}$; $\mathcal{R} \leftarrow \mathcal{R} \cup \{EZ, EO\}$;

For an encryption query $(T^s; P_1^s, P_2^s, \ldots, P_{m^s}^s; ty^s)$ respond as follows:
      $C_1^s||C_2^s||\ldots||C_{m^s}^s||\tau^s \xleftarrow{\$} \{0,1\}^{(m+1)n}$;
      Return $C_1^s||C_2^s||\ldots||C_{m^s}^s||\tau^s$;

Finalization:
FIRST PHASE
    for $s = 1$ to $q$,
      $\gamma^s \leftarrow h \cdot BRW(P_1^s||P_2^s||\ldots||P_{m^s}^s||T^s)$; $\tau^s \xleftarrow{\$} \{0,1\}^n$;
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{\gamma^s\}$; $\mathcal{R} \leftarrow \mathcal{R} \cup \{\tau^s\}$;
      for $i = 1$ to $m$,
        if $ty^s = L$ then $\delta_i^s = xP_i^s$
        else $\delta_i^s = (x+1)P_i^s$;
        end if
        $\mathcal{S} \leftarrow \mathcal{S} \cup \{\tau^s \oplus x^i EO\}$; $\mathcal{R} \leftarrow \mathcal{R} \cup \{C_i^s \oplus \delta_i^s\}$
      end for
    end for

SECOND PHASE
    bad = false;
    if (some value occurs more than once in $\mathcal{S}$) then bad = true endif;
    if (some value occurs more than once in $\mathcal{R}$) then bad = true endif.
```

can be detected by $\mathcal{B}$ only if the queries of $\mathcal{B}$ can produce a collision in the domain or range of the permutation π. This means that the difference can be detected by $\mathcal{B}$ if there is a collision in the multisets $\mathcal{S}$ or $\mathcal{R}$ as represented in equations (28) and (29) respectively. The event of a collision in these sets were represented by COLLD and COLLR respectively. Thus we have

$$\Pr[\mathcal{B}^{f_{h,\pi}(.),\mathcal{P}(.,.)} \text{ forges}] - \Pr[\mathcal{B}^{f_{h,\pi}(.),\$(.,.)} \text{ forges}] \leq \Pr[\mathsf{COLLD}] + \Pr[\mathsf{COLLR}]. \tag{32}$$

Now, the oracle $\$(.,.)$ is of no help to $\mathcal{B}$ as the random strings that it returns can be generated by $\mathcal{B}$ itself hence,

$$\Pr[\mathcal{B}^{f_{h,\pi}(.),\$(.,.)} \text{ forges}] = \Pr[\mathcal{B}^{f_{h,\pi}(.)} \text{ forges}]. \tag{33}$$

Putting together equations (32), (33) and (30) we have

$$\begin{aligned}\Pr[\mathcal{B}^{f_{h,\pi}(.),\mathcal{P}(.,.)} \text{ forges}] &\leq \Pr[\mathcal{B}^{f_{h,\pi}(.)} \text{ forges}] + \frac{14m^2q^2}{2^n}\\ &< \frac{1}{2^n} + \frac{4mq^2}{2^n} + \frac{14m^2q^2}{2^n}.\end{aligned} \tag{34}$$

The last inequality follows from Lemma 2 and eq. (1). From eq. (31) and eq. (34) we have

$$\mathbf{Adv}^{\text{dcm-auth}}_{\text{DCM-BRW}[\Pi]}(\mathcal{A}) \leq \frac{1}{2^n} + \frac{18m^2q^2}{2^n},$$

as desired. ■